

#10

JAN 17 2003

TRANSMITTAL OF FORMAL DRAWINGS

Docket No.
72478-0200DWB
1-2403

Re Application Of: Waki et al.

Serial No.	Filing Date	Batch No.	Examiner	Art Unit
09/288,263	April 8, 1999		Christian La Forgia	2157

Invention: HIGH SPEED VIRTUAL MACHINE AND COMPILER

Address to:
Assistant Commissioner for Patents
Washington, D.C. 20231

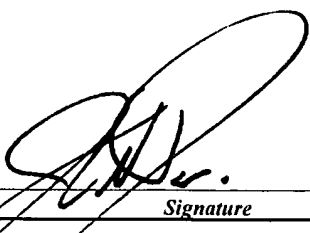
RECEIVED

JAN 21 2003

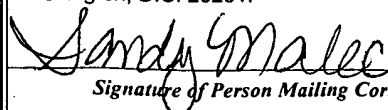
Technology Center 2100

Transmitted herewith are:

91 sheets of formal drawing(s) for this application.

☒ Each sheet of drawing indicates the identifying indicia suggested in 37 CFR Section 1.84(c).
Signature
Joseph W. Price, Esq.
Registration No. 25,124
Snell & Wilmer LLP
1920 Main Street, Suite 1200
Irvine, CA 92614
(949) 253-2700

Dated: 1-14-03

I certify that this document and attached formal drawings
are being deposited on 1-14-03 with the
U.S. Postal Service as first class mail under 37 C.F.R. 1.8
and addressed to the Assistant Commissioner for Patents,
Washington, D.C. 20231.
Signature of Person Mailing Correspondence

Sandy Malec

Typed or Printed Name of Person Mailing Correspondence

FIG. 1

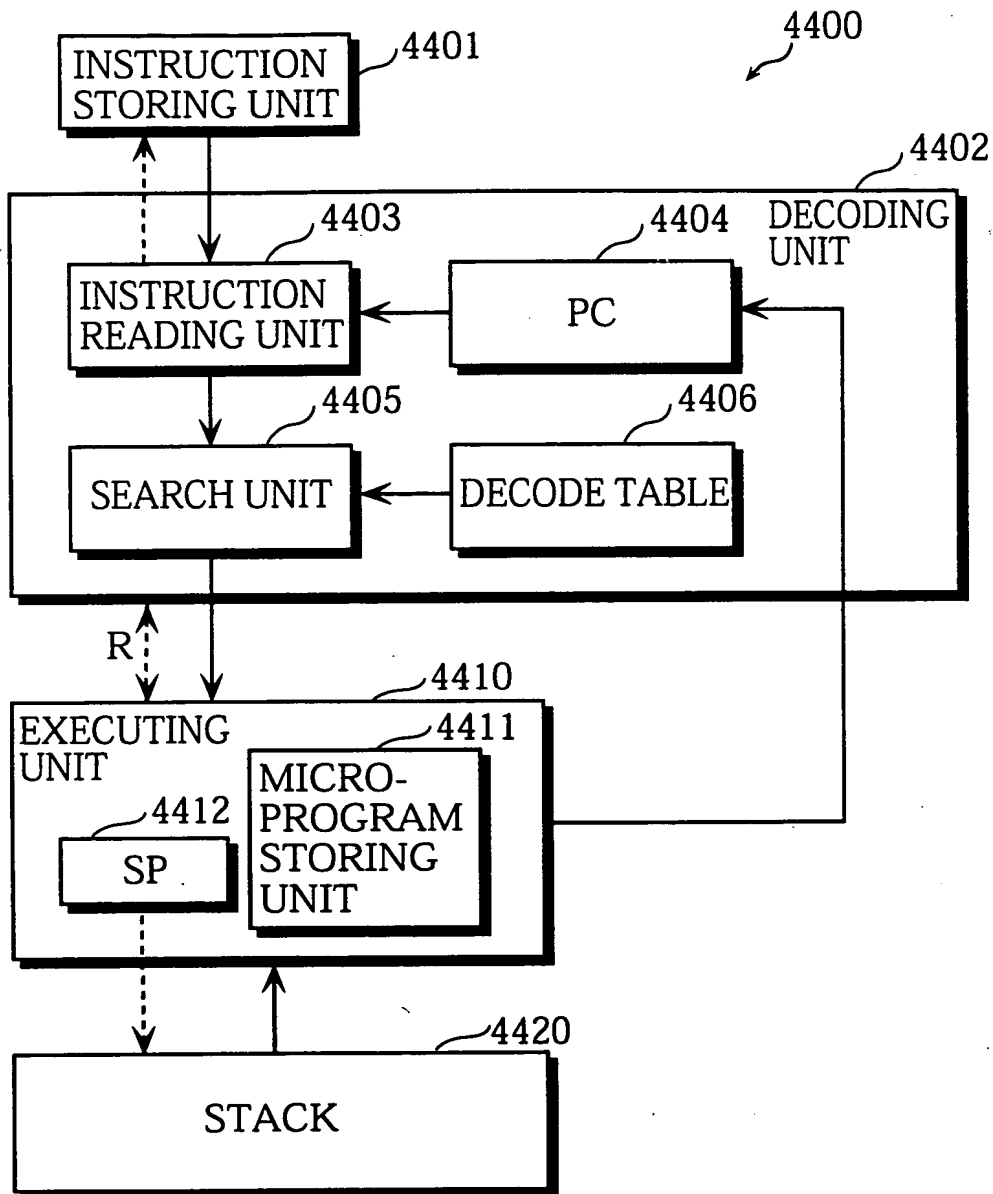


FIG. 2

VIRTUAL MACHINE INSTRUCTION	OPERATION CONTENTS	CHANGE IN STACK CONTENTS	SP VALUE
Push	PUSH OPERAND ONTO STACK	$s0 \leftarrow \text{operand}$	$sp \leftarrow sp+1$
Pop	POP VALUE OF TOP STACK AND PLACE IT INTO ADDRESS INDICATED BY OPERAND	$\text{operand} \leftarrow s0$ $s0 \leftarrow s1$	$sp \leftarrow sp-1$
Add	ADD VALUES ON TOP AND SECOND STACKS AND STORE RESULT ONTO STACK TOP	$s0 \leftarrow s0+s1$	$sp \leftarrow sp-1$
Mult	MULTIPLY VALUES ON TOP AND SECOND STACKS AND STORE RESULT ONTO STACK TOP	$s0 \leftarrow s0*s1$	$sp \leftarrow sp-1$
Br	JUMP UNCONDITIONALLY TO ADDRESS OF OPERAND	$s0 \leftarrow s0$	$sp \leftarrow sp$
Brz	JUMP TO ADDRESS OF OPERAND IF STACK TOP VALUE IS 0	DELETE $\leftarrow s0$ $s0 \leftarrow s1$	$sp \leftarrow sp-1$
Brnz	JUMP TO ADDRESS OF OPERAND IF STACK TOP VALUE IS NOT 0	DELETE $\leftarrow s0$ $s0 \leftarrow s1$	$sp \leftarrow sp-1$
Call	CALL FUNCTION SPECIFIED BY ADDRESS OF OPERAND	$s0 \leftarrow \text{NEXT VIRTUAL MACHINE CODE ADDRESS}$	$sp \leftarrow sp+1$
Ret	JUMP UNCONDITIONALLY TO ADDRESS OF STACK TOP VALUE	DELETE $\leftarrow s0$ $s0 \leftarrow s1$	$sp \leftarrow sp-1$
Stop	STOP VIRTUAL MACHINE PROCESSING	INITIAL STATE	$sp \leftarrow 0$

FIG. 3

4406a OPCODE	4406b JUMP ADDRESS	4406c NUMBER OF OPERANDS
:	:	:
Push	<JUMP ADDRESS OF CODE TO PERFORM Push>	1
Pop	<JUMP ADDRESS OF CODE TO PERFORM Pop>	1
Add	<JUMP ADDRESS OF CODE TO PERFORM Add>	0
Sub	<JUMP ADDRESS OF CODE TO PERFORM Sub>	0
Inc	<JUMP ADDRESS OF CODE TO PERFORM Inc>	0
Dec	<JUMP ADDRESS OF CODE TO PERFORM Dec>	0
Mult	<JUMP ADDRESS OF CODE TO PERFORM Mult>	0
Div	<JUMP ADDRESS OF CODE TO PERFORM Div>	0
:	:	:



FIG. 4A

MICROPROGRAM FOR VIRTUAL MACHINE INSTRUCTION "Push"		
1:Inc	r3	; INCREMENT SP VALUE BY ONE
2:Load	r0,[r2]	; EXTRACT OPERAND AND ; PLACE IT ONTO REGISTER #0
3:Inc	r2	; INCREMENT VIRTUAL MACHINE PC BY ONE AND ; PREPARE FOR READING NEXT INSTRUCTION
4:Store	[r3],r0	; PUSH VALUE OF REGISTER #0 ONTO STACK
<MICROPROGRAM FOR JUMPING TO NEXT VIRTUAL MACHINE INSTRUCTION>		

FIG. 4B

MICROPROGRAM FOR VIRTUAL MACHINE INSTRUCTION "Add"		
1:Load	r0,[r3]	; EXTRACT VALUE FROM STACK ; PLACE IT ONTO REGISTER #0
2:Dec	r3	; DECREMENT VALUE OF VIRTUAL MACHINE SP BY ONE
3:Load	r1,[r3]	; EXTRACT VALUE FROM STACK ; PLACE IT ONTO REGISTER #1
4:Add	r0,r0,r1	; ADD VALUES OF REGISTER #0 AND #1 AND ; PLACE RESULT ONTO REGISTER #1
5:Store	[r3],r0	; PLACE VALUE OF REGISTER #0 ONTO STACK
<MICROPROGRAM FOR JUMPING TO NEXT VIRTUAL MACHINE INSTRUCTION>		

FIG. 4C

MICROPROGRAM FOR VIRTUAL MACHINE INSTRUCTION "Mult"		
1:Load	r0,[r3]	; EXTRACT VALUE FROM STACK AND ; PLACE IT ONTO REGISTER #0
2:Dec	r3	; DECREMENT VALUE OF VIRTUAL MACHINE SP BY ONE
3:Load	r1,[r3]	; EXTRACT VALUE FROM STACK AND ; PLACE IT ONTO REGISTER #1
4:Mult	r0,r0,r1	; MULTIPLY VALUES OF REGISTERS #0 AND #1 AND ; PLACE RESULT ONTO REGISTER #1
5:Store	[r3],r0	; PLACE VALUE OF REGISTER #0 ONTO STACK
<MICROPROGRAM FOR JUMPING TO NEXT VIRTUAL MACHINE INSTRUCTION>		

FIG. 4D

<MICROPROGRAM FOR JUMPING TO NEXT VIRTUAL MACHINE INSTRUCTION>		
1:Load	r0,[r2]	; READ VIRTUAL MACHINE INSTRUCTION (JUMP ADDRESS) INDICATED BY PC INTO REGISTER #0
2:Inc	r2	; INCREMENT VIRTUAL MACHINE PC VALUE BY ONE
3:Jmp	r0	; JUMP UNCONDITIONALLY TO LOCATION INDICATED BY REGISTER #0

FIG. 5

REAL MACHINE INSTRUCTION	OPERATION CONTENTS	NOTATION	EXAMPLE
Load	PLACE VALUE OF DESIGNATED REGISTER (x1) OR THAT OF MEMORY LOCATION ([x1]) DESIGNATED BY REGISTER ONTO VALUE OF DESIGNATED REGISTER (x0)	Load x0,x1 Load x0,[x1]	Load r0,r1 Load r1,[r2]
Store	PLACE VALUE OF DESIGNATED REGISTER (x1) ONTO MEMORY LOCATION [x0] DESIGNATED BY REGISTER	Store [x0],x1	Store [r0],r1
Add	ADD VALUES OF DESIGNATED REGISTERS (x1,x2), AND PLACE RESULT ONTO DESIGNATED REGISTER (x0)	Add x0,x1,x2	Add r0,r1,r2
Mult	MULTIPLY VALUES OF DESIGNATED REGISTERS (x1,x2), AND PLACE RESULT ONTO DESIGNATED REGISTER (x0)	Mult x0,x1,x2	Mult r0,r1,r2
Inc	ADD 1 TO VALUE OF DESIGNATED REGISTER (x1), AND STORE RESULT IN SAME REGISTER	Inc x0	Inc r4
Dec	SUBTRACT 1 FROM VALUE OF DESIGNATED REGISTER (x0), AND STORE RESULT IN SAME REGISTER	Dec x0	Dec r3
Jump	JUMP TO ADDRESS INDICATED BY DESIGNATED REGISTER (x0)	Jump x0	Jump r2
Jz	JUMP TO ADDRESS INDICATED BY DESIGNATED REGISTER (x1) IF VALUE OF DESIGNATED REGISTER (x0) IS 0	Jz x0,x1	Jz r0,r2
Jnz	JUMP TO ADDRESS INDICATED BY DESIGNATED REGISTER (x1) IF VALUE OF DESIGNATED REGISTER (x0) IS NOT 0	Jnz x0,x1	Jnz r0,r2
Nop	NOTHING PERFORMED	Nop	Nop

FIG. 6

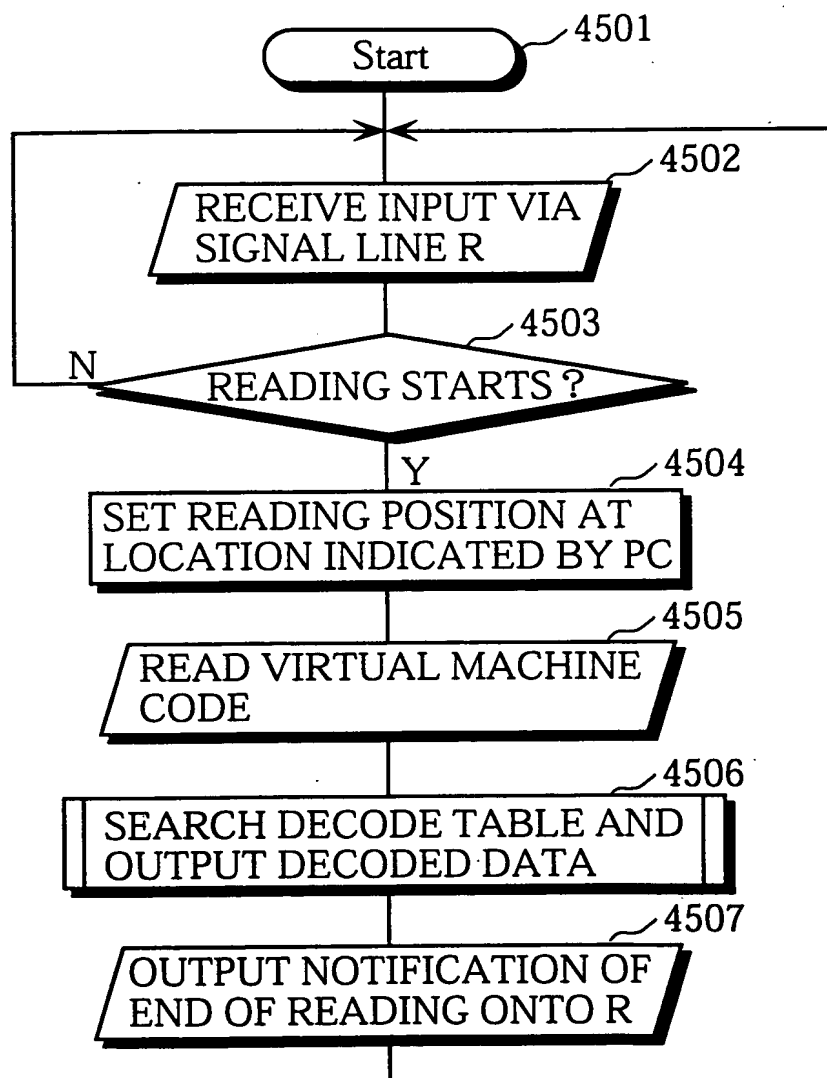


FIG. 7

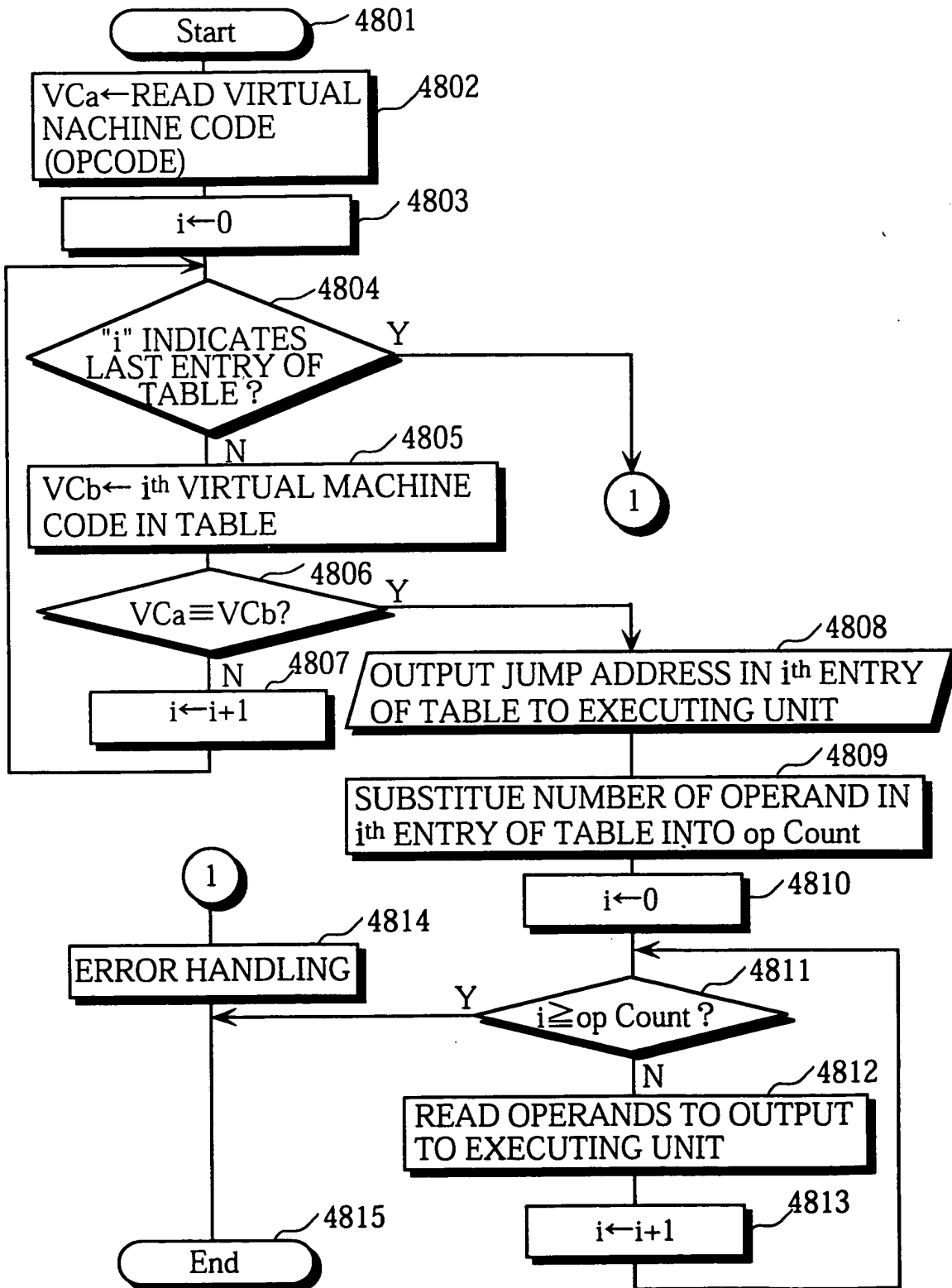
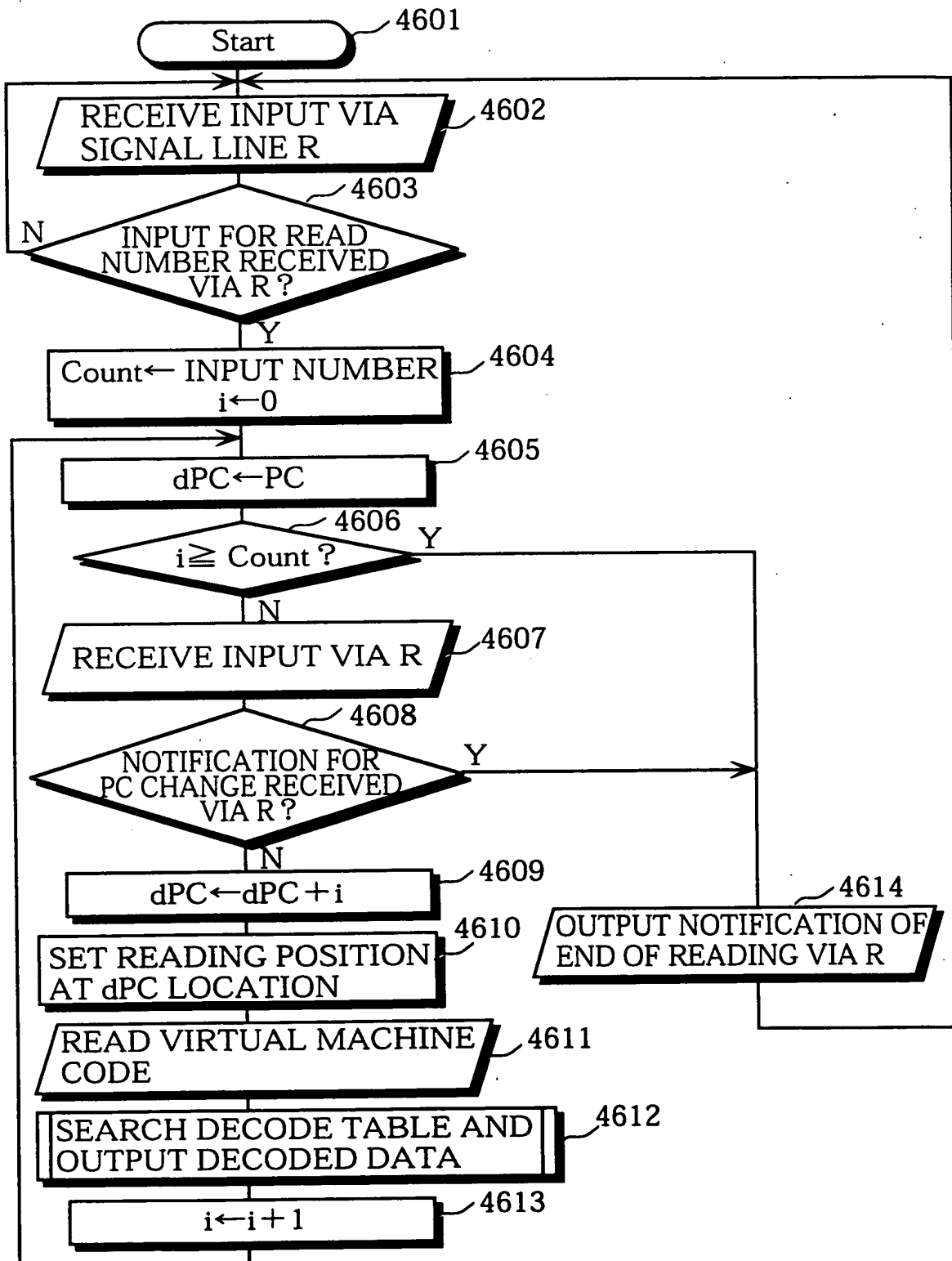


FIG. 8



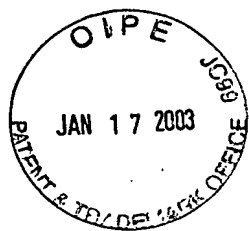


FIG. 9

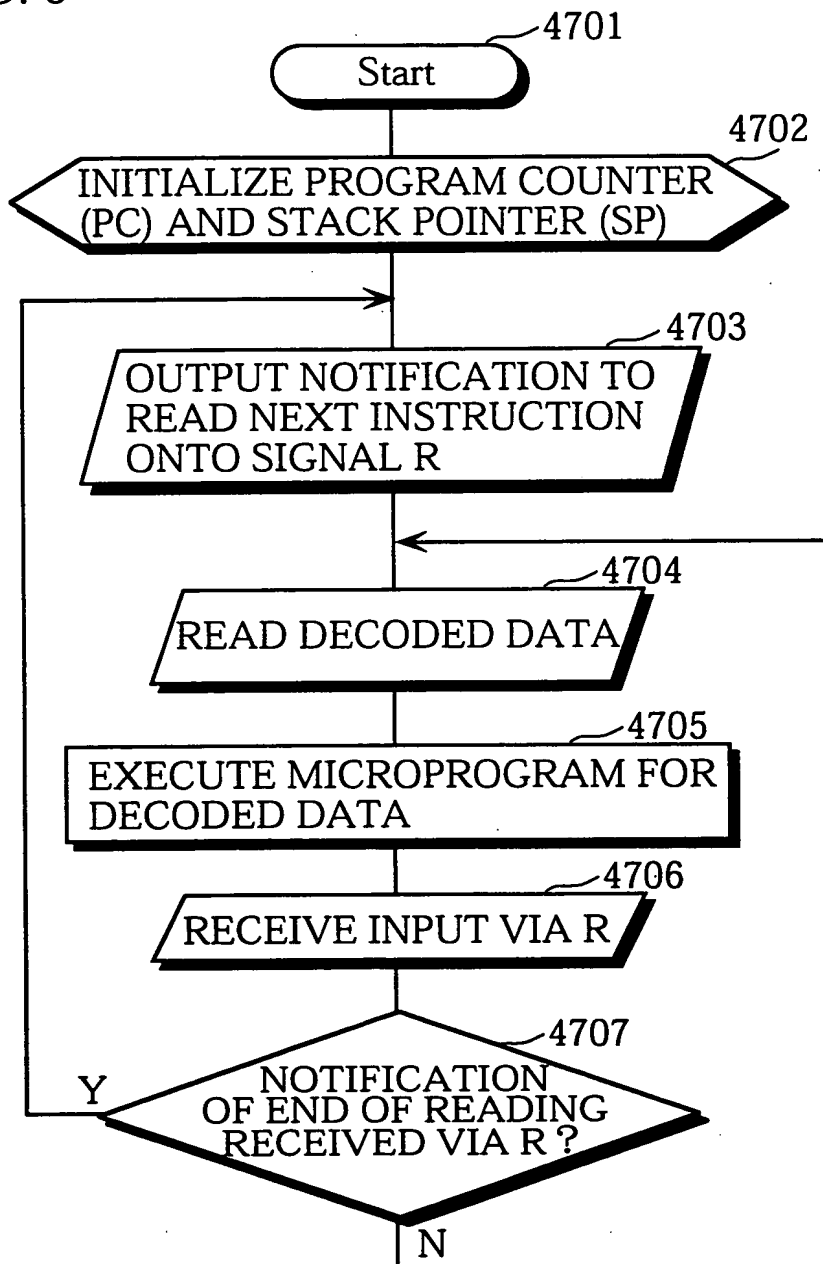




FIG. 10A

1:	Push
2:	2
3:	Push
4:	3
5:	Push
6:	4
7:	Add
8:	Mult
9:	Pop
10:	0

FIG. 10B

ARITHMETIC EXPRESSION : $\langle \text{DATA AREA \#0} \rangle = 2 * (3 + 4)$

FIG. 10C

1:	$\langle \text{JUMP ADDRESS OF CODE TO PERFORM Push} \rangle$
2:	OPERAND "2"
3:	$\langle \text{JUMP ADDRESS OF CODE TO PERFORM Push} \rangle$
4:	OPERAND "3"
5:	$\langle \text{JUMP ADDRESS OF CODE TO PERFORM Push} \rangle$
6:	OPERAND "4"
7:	$\langle \text{JUMP ADDRESS OF CODE TO PERFORM Add} \rangle$
8:	$\langle \text{JUMP ADDRESS OF CODE TO PERFORM Mult} \rangle$
9:	$\langle \text{JUMP ADDRESS OF CODE TO PERFORM Pop} \rangle$
10:	OPERAND "0"

FIG. 11A

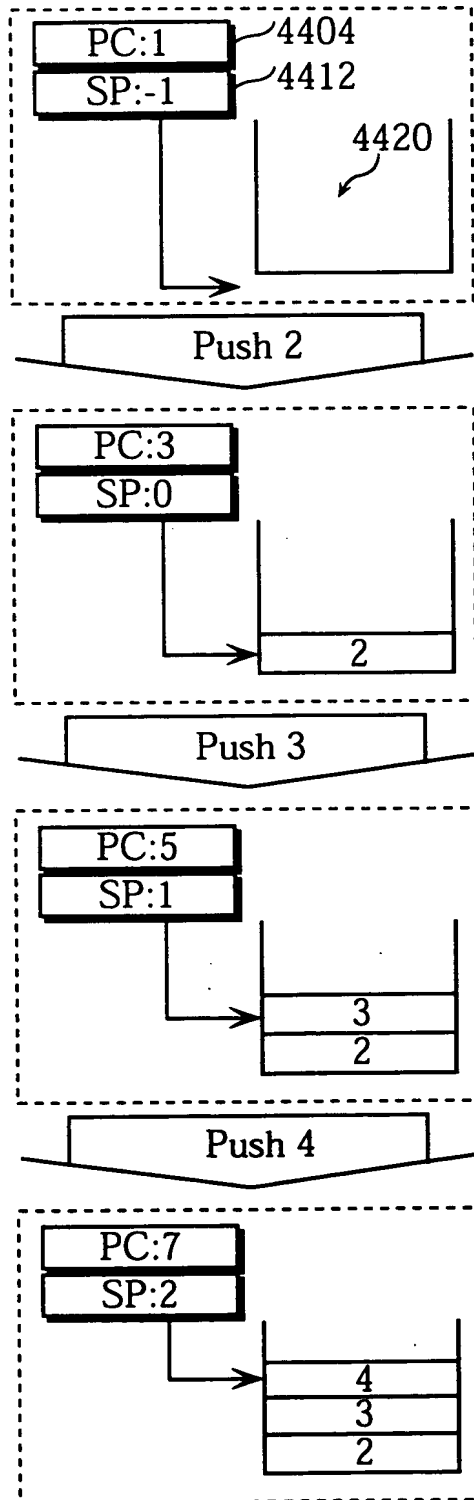


FIG. 11B

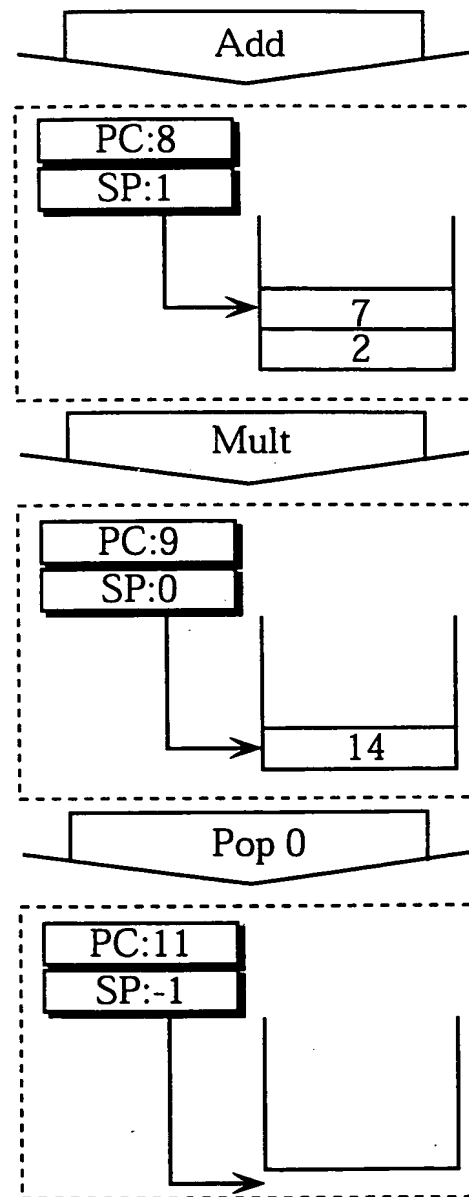




FIG. 12A

MICROPROGRAM FOR VIRTUAL MACHINE INSTRUCTION "Push"		
1:Inc	r3	; INCREMENT SP VALUE BY ONE
2:Store	[r3],r0	; PLACE VALUE OF TOS REGISTER (#0) ; INTO STACK
3:Load	r0,[r2]	; EXTRACT OPERAND AND ; PLACE IT ONTO TOS REGISTER
4:Inc	r2	; INCREMENT PC OF VIRTUAL MACHINE BY ONE TO PREPARE ; FOR READING NEXT INSTRUCTION
<MICROPROGRAM FOR JUMPING TO NEXT VIRTUAL MACHINE INSTRUCTION>		

FIG. 12B

MICROPROGRAM FOR VIRTUAL MACHINE INSTRUCTION "Add"		
1:Load	r1,[r3]	; EXTRACT VALUE FROM STACK ; PLACE IT ONTO REGISTER #1
2:Dec	r3	; DECREMENT VALUE OF VIRTUAL MACHINE PC BY ONE
3:Add	r0,r0,r1	; ADD VALUES OF REGISTERS #0 AND #1 AND ; PLACE RESULT ONTO TOS REGISTER
<MICROPROGRAM FOR JUMPING TO NEXT VIRTUAL MACHINE INSTRUCTION>		

FIG. 12C

MICROPROGRAM FOR VIRTUAL MACHINE INSTRUCTION "Mult"		
1:Load	r1,[r3]	; EXTRACT VALUE FROM STACK AND ; PLACE IT ONTO REGISTER #1
2:Dec	r3	; DECREMENT VALUE OF VIRTUAL MACHINE SP BY ONE
3:Mult	r0,r0,r1	; MULTIPLY VALUES OF REGISTERS #0 AND #1 AND ; PLACE RESULT ONTO TOS REGISTER
<MICROPROGRAM FOR JUMPING TO NEXT VIRTUAL MACHINE INSTRUCTION>		

FIG. 12D

<MICROPROGRAM FOR JUMPING TO NEXT VIRTUAL MACHINE INSTRUCTION>		
1:Load	r1,[r2]	; READ VIRTUAL MACHINE INSTRUCTION (JUMP ADDRESS) ; INDICATED BY PC INTO REGISTER #1
2:Inc	r2	; INCREMENT VIRTUAL MACHINE PC BY ONE
3:Jmp	r1	; JUMP UNCONDITIONALLY TO LOCATION INDICATED BY REGISTER #1

FIG. 13A

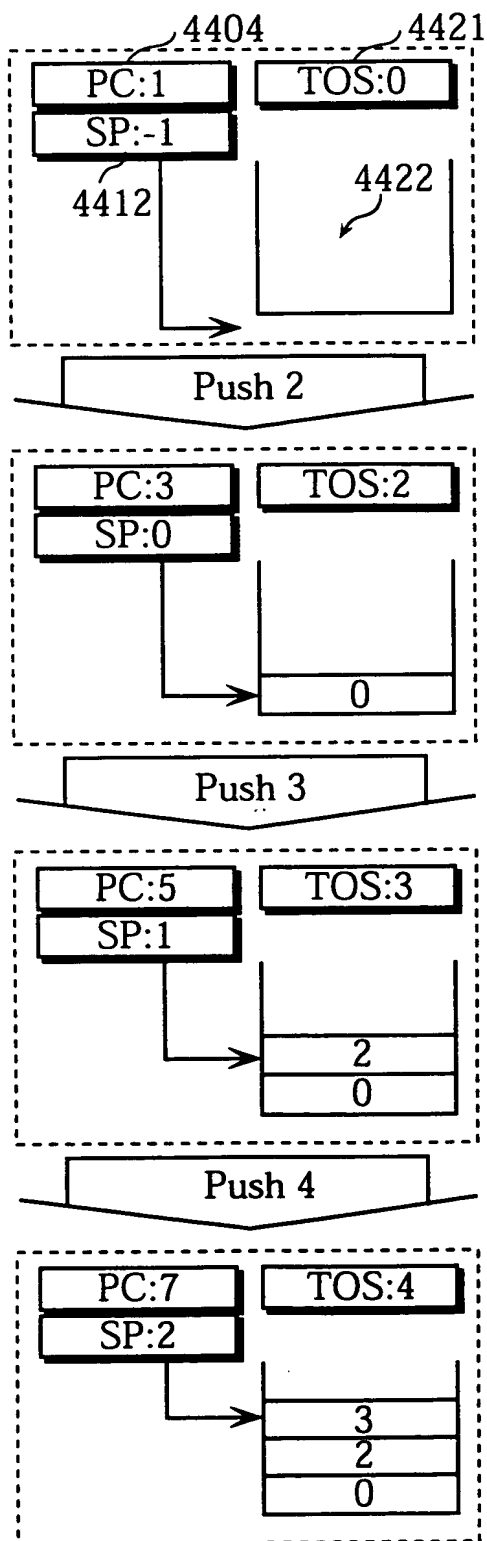


FIG. 13B

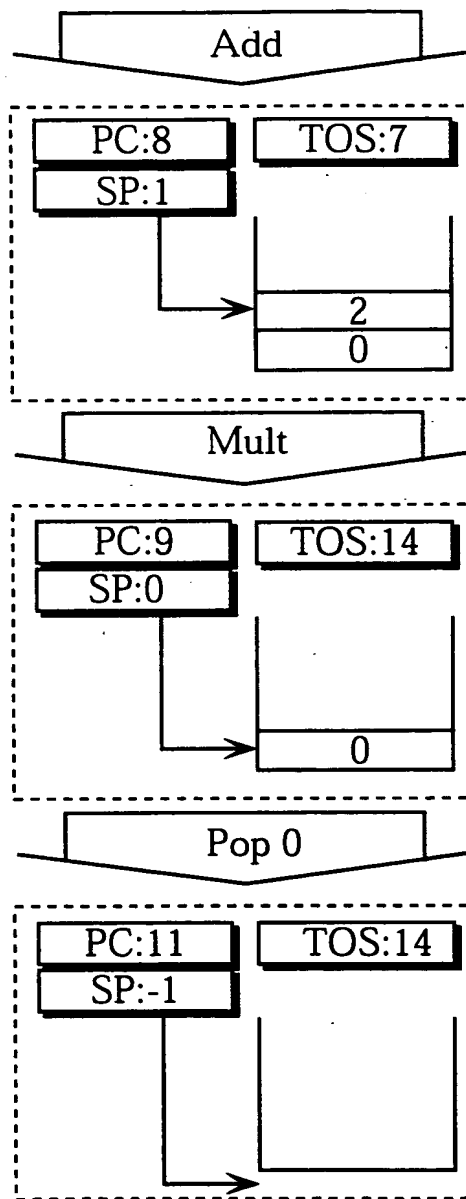




FIG. 14

STAGE NAME	NOTATION
INSTRUCTION FETCH	IF
INSTRUCTION DECODE AND REGISTER REFERENCE	RF
EXECUTION	ALU
MEMORY ACCESS	MEM
RESULT WRITING TO REGISTER	WB

FIG. 15

CLOCK	1	2	3	4	5	6	7	8
INSTRUCTION A	IF	RF	ALU	MEM	WB			
INSTRUCTION B		IF	RF	ALU	MEM	WB		
INSTRUCTION C			IF	RF	ALU	MEM	WB	
INSTRUCTION D				IF	RF	ALU	MEM	WB

FIG. 16

CLOCK	1	2	3	4	5	6	7	8
INSTRUCTION A1	IF	RF	ALU	MEM	WB			
INSTRUCTION A2	IF	RF	ALU	MEM	WB			
INSTRUCTION B1		IF	RF	ALU	MEM	WB		
INSTRUCTION B2		IF	RF	ALU	MEM	WB		
INSTRUCTION C1			IF	RF	ALU	MEM	WB	
INSTRUCTION C2			IF	RF	ALU	MEM	WB	
INSTRUCTION D1				IF	RF	ALU	MEM	WB
INSTRUCTION D2				IF	RF	ALU	MEM	WB



FIG. 17

CLOCK	1	2	3	4	5	6	7	8	9
INSTRUCTION A	IF	RF	ALU	MEM	WB				
INSTRUCTION B		IF	RF	•	ALU	MEM	WB		
INSTRUCTION C			IF	•	RF	ALU	MEM	WB	
INSTRUCTION D				•	IF	RF	ALU	MEM	WB

FIG. 18

CLOCK	1	2	3	4	5	6	7	8	9
INSTRUCTION A1	IF	RF	ALU	MEM	WB				
INSTRUCTION A2	IF	RF	ALU	MEM	WB				
INSTRUCTION B1		IF	RF	•	ALU	MEM	WB		
INSTRUCTION B2		IF	RF	ALU	MEM	WB			
INSTRUCTION C1			IF	RF	ALU	MEM	WB		
INSTRUCTION C2			IF	RF	•	ALU	MEM	WB	
INSTRUCTION D1				IF	RF	ALU	MEM	WB	
INSTRUCTION D2				IF	RF	•	ALU	MEM	WB

FIG. 19

CLOCK	1	2	3	4	5	6	7	8	9	10
INSTRUCTION A	IF	RF	ALU	MEM	WB					
INSTRUCTION B		IF	RF	•	•	ALU	MEM	WB		
INSTRUCTION C			IF	•	•	RF	ALU	MEM	WB	
INSTRUCTION D				•	•	IF	RF	ALU	MEM	WB



FIG. 20

CLOCK	1	2	3	4	5	6	7	8	9
INSTRUCTION A1	IF	RF	ALU	MEM	WB				
INSTRUCTION A2	IF	RF	ALU	MEM	WB				
INSTRUCTION B1		IF	RF	•	•	ALU	MEM	WB	
INSTRUCTION B2		IF	RF	ALU	MEM	WB			
INSTRUCTION C1			IF	RF	ALU	MEM	WB		
INSTRUCTION C2			IF	RF	ALU	MEM	WB		
INSTRUCTION D1				IF	RF	ALU	MEM	WB	
INSTRUCTION D2				IF	RF	•	ALU	MEM	WB

FIG. 21

CLOCK	1	2	3	4	5	6	7
INSTRUCTION A	IF	RF	ALU	MEM	WB		
INSTRUCTION B		IF	x				
INSTRUCTION C			IF	RF	ALU	MEM	WB

FIG. 22

CLOCK	1	2	3	4	5	6	7
INSTRUCTION A1	IF	RF	ALU	MEM	WB		
INSTRUCTION A2	IF	RF	ALU	MEM	WB		
INSTRUCTION B1		IF	x				
INSTRUCTION B2		IF	x				
INSTRUCTION C1			IF	RF	ALU	MEM	WB
INSTRUCTION C2			IF	RF	ALU	MEM	WB

FIG. 23

CLOCK	1	2	3	4	5	6	7	8	9	10	11
Load r1,[r2]	IF	RF	ALU	MEM	WB						
Inc r2		IF	RF	ALU	MEM	WB					
Jmp r1			IF	RF	ALU	MEM	WB				
				IF	x						
Load r1,[r3]					IF	RF	ALU	MEM	WB		
Dec r3						IF	RF	ALU	MEM	WB	
Mult r0,r0,r1							IF	RF	ALU	MEM	WB

FIG. 24

CLOCK	1	2	3	4	5	6	7	8	9	10	11
Load r1,[r2]	IF	RF	ALU	MEM	WB						
Inc r2	IF	RF	ALU	MEM	WB						
Jmp r1		IF	RF	•	ALU	MEM	WB				
		IF	RF	ALU	x						
			IF	RF	x						
			IF	RF	x						
				IF	x						
				IF	x						
Load r1,[r3]					IF	RF	ALU	MEM	WB		
Dec r3					IF	RF	ALU	MEM	WB		
Mult r0,r0,r1						IF	RF	•	ALU	MEM	WB



FIG. 25

CLOCK	1	2	3	4	5	6	7	8	9	10	11	12	13
Load r1,[r2]	IF	RF	ALU	MEM	WB								
Inc r2		IF	RF	ALU	MEM	WB							
Jmp r1			IF	RF	•	ALU	MEM	WB					
				IF	•	x							
Load r1,[r3]					•	IF	RF	ALU	MEM	WB			
Dec r3							IF	RF	ALU	MEM	WB		
Mult r0,r0,r1								IF	RF	•	ALU	MEM	WB



FIG. 26

CLOCK	1	2	3	4	5	6	7	8	9	10	11	12	13
Load r1,[r2]	IF	RF	ALU	MEM	WB								
Inc r2	IF	RF	ALU	MEM	WB								
Jmp r1		IF	RF	•	•	ALU	MEM	WB					
		IF	RF	ALU	MEM	x							
			IF	RF	ALU	x							
			IF	RF	ALU	x							
				IF	RF	x							
				IF	RF	x							
					IF	x							
					IF	x							
Load r1,[r3]						IF	RF	ALU	MEM	WB			
Dec r3						IF	RF	ALU	MEM	WB			
Mult r0,r0,r1							IF	RF	•	•	ALU	MEM	WB



FIG. 27

0:Push	0	;i←0
2:Pop	[0]	
4:Push	0	;sum←0
6:Pop	[1]	
8:Push	[0]	;i<10?
10:Push	10	
12:Sub		
13:Brz	31	
15:Push	[1]	;sum←sum+i
17:Push	[0]	
19:Add		
20:Pop	[1]	
22:Push	[0]	;i←i+1
24:Push	1	
26:Add		
27:Pop	[0]	
29:Br	8	
31:Stop		



FIG. 28

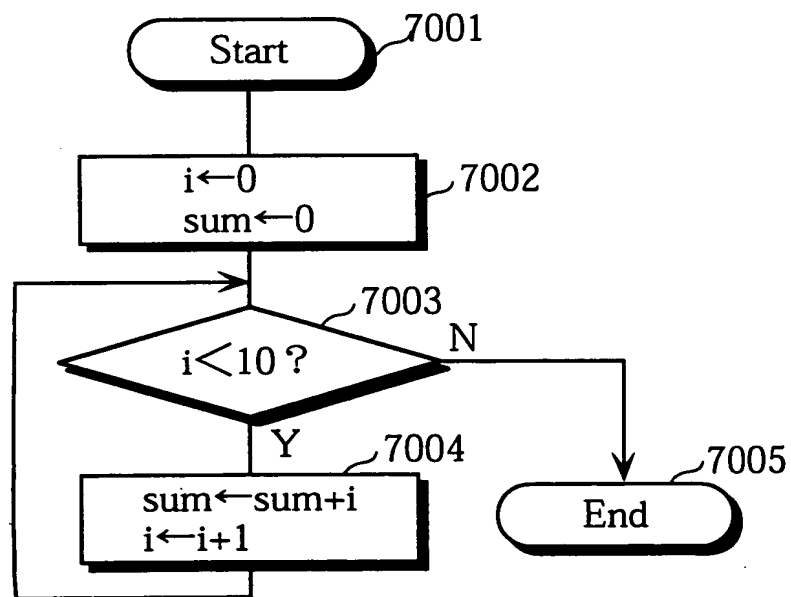




FIG. 29

VIRTUAL MACHINE INSTRUCTION	SIZE OF REAL MACHINE CODE TEMPLATE	REAL MACHINE CODE TEMPLATE	NUMBER OF OPERANDS
:	:	:	:
Push	4 WORDS	<CODE TO PERFORM Push>	1
Pop	5 WORDS	<CODE TO PERFORM Pop>	1
Add	3 WORDS	<CODE TO PERFORM Add>	0
Sub	3 WORDS	<CODE TO PERFORM Sub>	0
Mult	3 WORDS	<CODE TO PERFORM Mult>	0
Push[]	5 WORDS	<CODE TO PERFORM Push[]>	1
Br	3 WORDS	<CODE TO PERFORM Br>	1
Brz	5 WORDS	<CODE TO PERFORM Brz>	1
Stop	2 WORDS	<CODE TO PERFORM Stop>	0
:	:	:	:



FIG. 30

VIRTUAL MACHINE CODE ADDRESS	VIRTUAL MACHINE CODE	REAL MACHINE CODE SIZE	CORRESPONDING REAL MACHINE CODE ADDRESS
0	Push 0	4	0-3
2	Pop [0]	5	4-8
4	Push 0	4	9-12
6	Pop [1]	5	13-17
8	Push [0]	5	18-22
10	Push 10	4	23-26
12	Sub	3	27-29
13	Brz 31	5	30-34
15	Push [1]	5	35-39
17	Push [0]	5	40-44
19	Add	3	45-47
20	Pop [1]	5	48-52
22	Push [0]	5	53-57
24	Push 1	4	58-61
26	Add	3	62-64
27	Pop [0]	5	65-69
29	Br 8	3	70-72
31	Stop	2	73-75

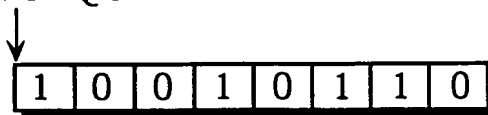


FIG. 31A

BIT SEQUENCE	MEANING
0	"a"
10	"b"
110	"c"
111	"d"

FIG. 31B

INSTRUCTION SEQUENCE A:"babc"



INSTRUCTION SEQUENCE B:"aabc"

FIG. 32

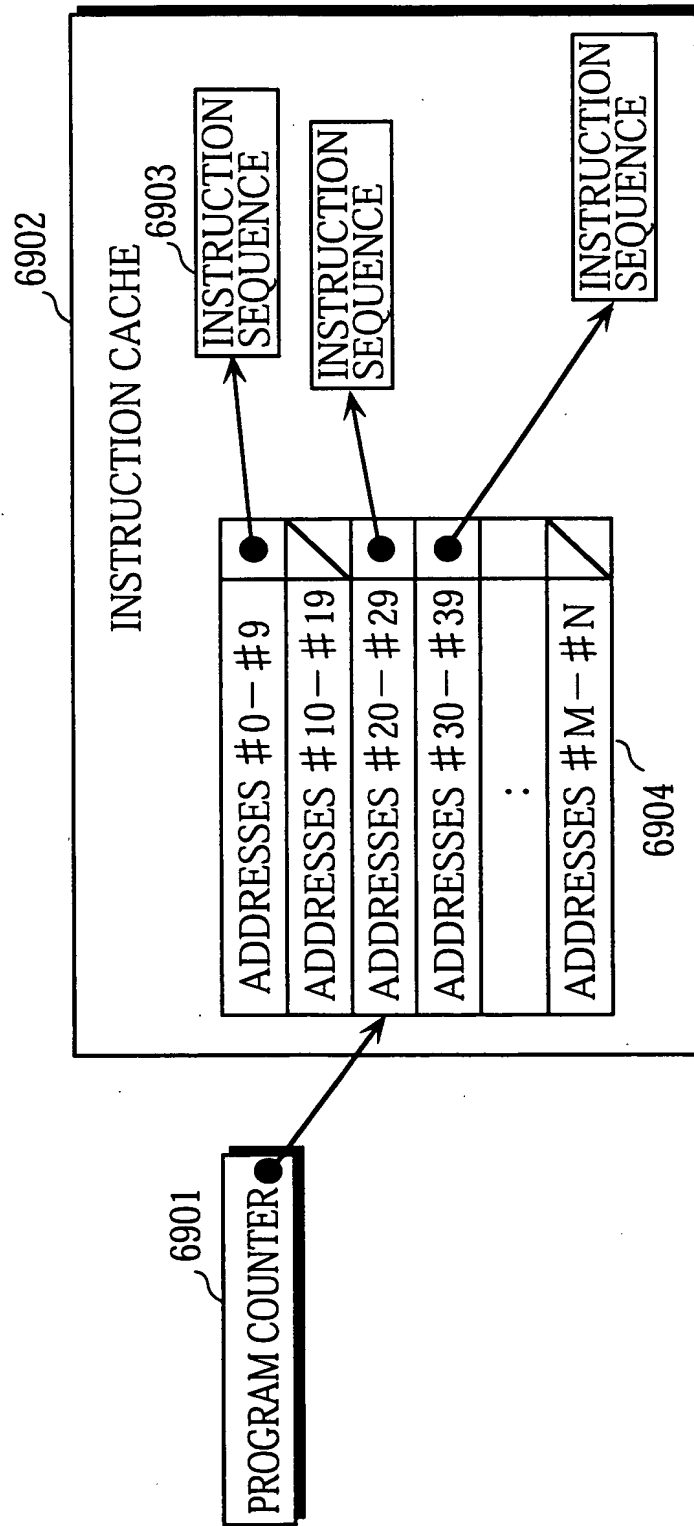




FIG. 33

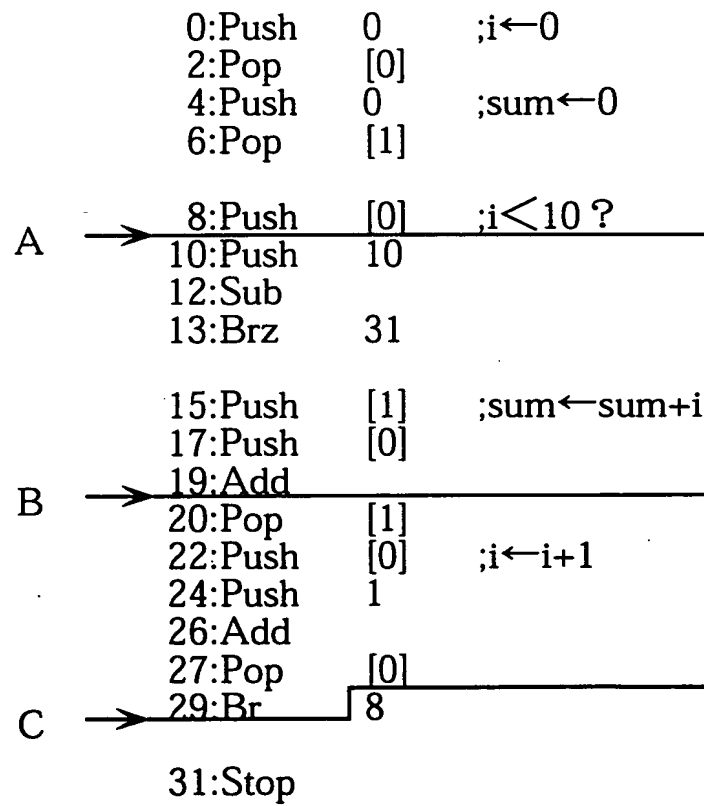


FIG. 34

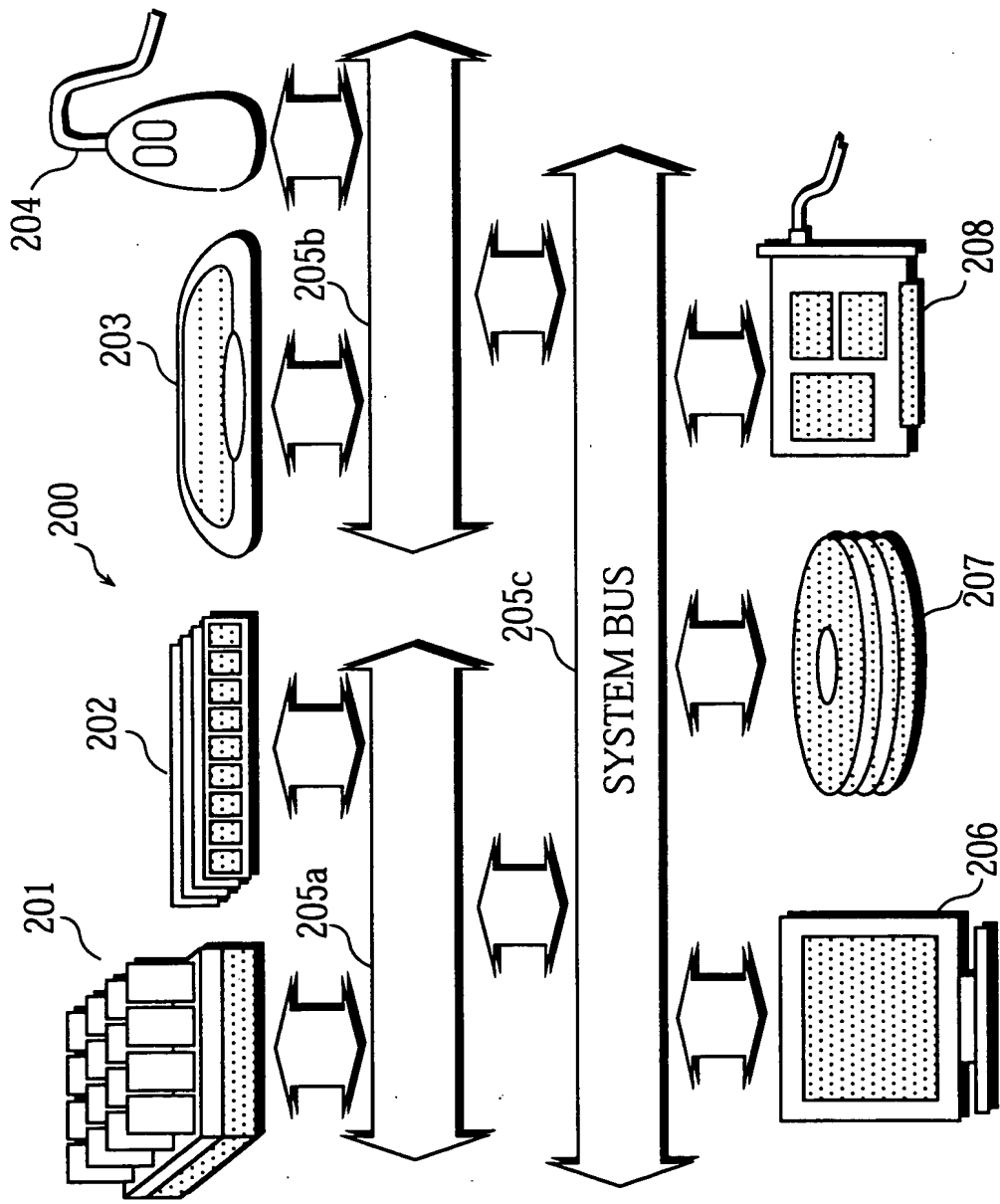


FIG. 35

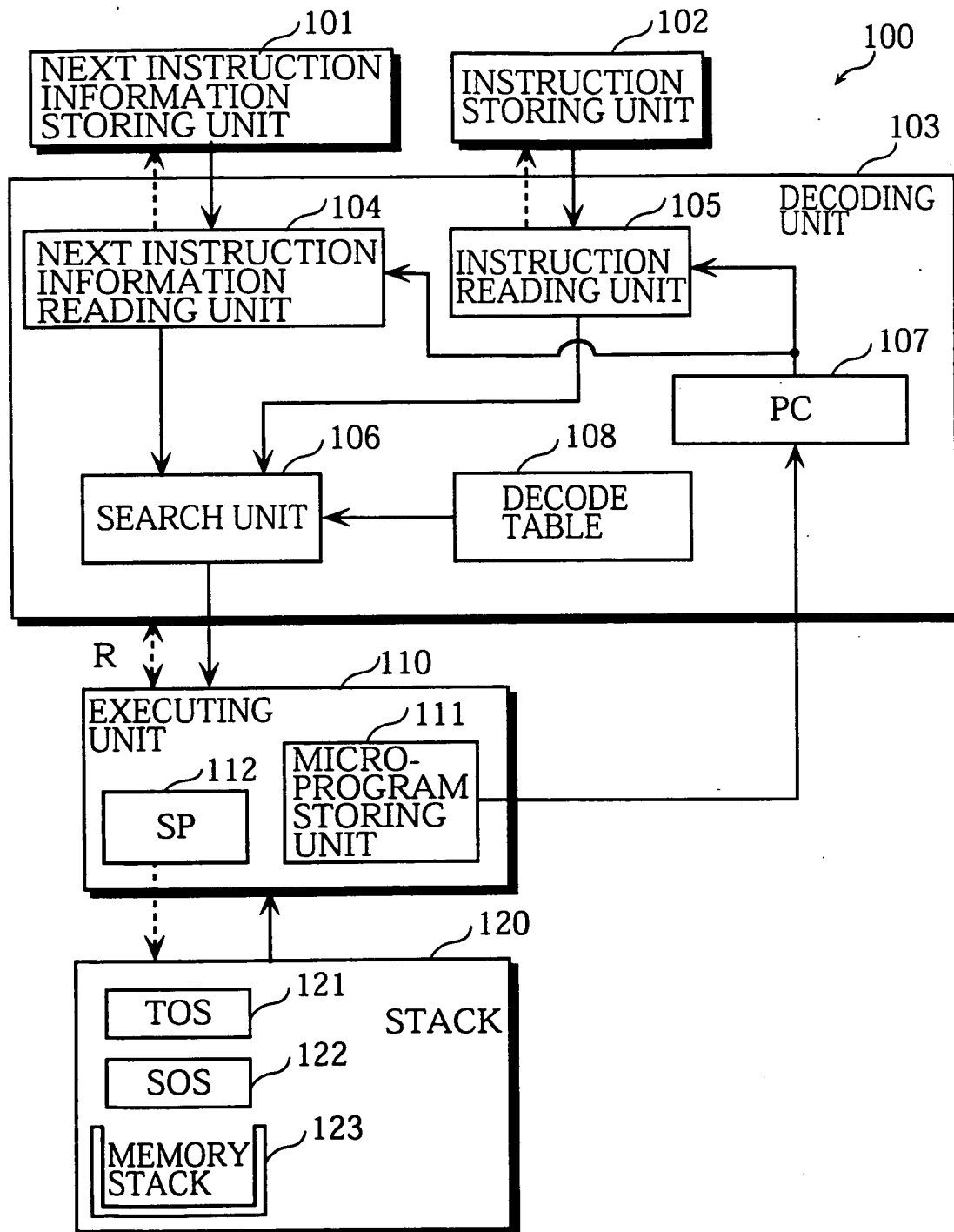




FIG. 36A

101

1:	U
2:	U
3:	U
4:	U
5:	D
6:	D
7:	D
8:	D
9:	U
10:	U

FIG. 36B

102

1:	Push
2:	2
3:	Push
4:	3
5:	Push
6:	4
7:	Add
8:	Mult
9:	Pop
10:	0

FIG. 37

OPCODE	NEXT INSTRUCTION INFORMATION	JUMP ADDRESS	NUMBER OF OPERANDS
:	:	:	:
Push	U	<JUMP ADDRESS OF CODE TO PERFORM Push ASSIGNED "U">	1
Push	D	<JUMP ADDRESS OF CODE TO PERFORM Push ASSIGNED "D">	1
Pop	U	<JUMP ADDRESS OF CODE TO PERFORM Pop ASSIGNED "U">	1
Pop	D	<JUMP ADDRESS OF CODE TO PERFORM Pop ASSIGNED "D">	1
Add	U	<JUMP ADDRESS OF CODE TO PERFORM Add ASSIGNED "U">	0
Add	D	<JUMP ADDRESS OF CODE TO PERFORM Add ASSIGNED "D">	0
Sub	U	<JUMP ADDRESS OF CODE TO PERFORM Sub ASSIGNED "U">	0
Sub	D	<JUMP ADDRESS OF CODE TO PERFORM Sub ASSIGNED "D">	0
Inc	U	<JUMP ADDRESS OF CODE TO PERFORM Inc ASSIGNED "U">	0
Inc	D	<JUMP ADDRESS OF CODE TO PERFORM Inc ASSIGNED "D">	0
:	:	:	:



FIG. 38A

MICROPROGRAM FOR VIRTUAL MACHINE INSTRUCTION "Push" WITH "U"		
1:Load	r4,r0	;COPY VALUE OF TOS REGISTER (#0) INTO SOS REGISTER (#4)
2:Load	r0,[r2]	;READ OPERAND INTO TOS REGISTER
3:Inc	r2	;INCREMENT VIRTUAL MACHINE PC BY ONE TO PREPARE FOR READING NEXT INSTRUCTION
4:Inc	r3	;INCREMENT VIRTUAL MACHINE SP BY ONE
5:Store	[r3],r4	;PLACE SOS REGISTER VALUE INTO STACK
6:Load	r1,[r2]	;READ VIRTUAL MACHINE INSTRUCTION (JUMP ADDRESS) INDICATED BY PC INTO REGISTER #1
7:Inc	r2	;INCREMENT VIRTUAL MACHINE PC BY ONE
8:Jmp	r1	;JUMP UNCONDITIONALLY TO LOCATION INDICATED BY REGISTER #1

FIG. 38B

MICROPROGRAM FOR VIRTUAL MACHINE INSTRUCTION "Push" WITH "D"		
1:Load	r4,r0	;COPY VALUE OF TOS REGISTER (#0) INTO SOS REGISTER (#4)
2:Load	r0,[r2]	;READ OPERAND INTO TOS REGISTER
3:Inc	r2	;INCREMENT VIRTUAL MACHINE PC BY ONE TO PREPARE FOR READING NEXT INSTRUCTION
6:Load	r1,[r2]	;READ VIRTUAL MACHINE INSTRUCTION (JUMP ADDRESS) INDICATED BY PC INTO REGISTER #1
7:Inc	r2	;INCREMENT VIRTUAL MACHINE PC BY ONE
8:Jmp	r1	;JUMP UNCONDITIONALLY TO LOCATION INDICATED BY REGISTER #1



FIG. 39A

MICROPROGRAM FOR VIRTUAL MACHINE INSTRUCTION "Add" WITH "U"	
1: Add	r0,r0,r4 ;ADD VALUES OF TOS REGISTER AND SOS REGISTER, AND PLACE RESULT INTO TOS REGISTER
<MICROPROGRAM FOR VIRTUAL MACHINE JUMP CODE WITH "U">	

FIG. 39B

MICROPROGRAM FOR VIRTUAL MACHINE INSTRUCTION "Add" WITH "D"	
1: Add	r0,r0,r4 ;ADD VALUES OF TOS REGISTER AND SOS REGISTER, AND PLACE RESULT INTO TOS REGISTER
<MICROPROGRAM FOR VIRTUAL MACHINE JUMP CODE WITH "D">	



FIG. 40A

MICROPROGRAM FOR VIRTUAL MACHINE INSTRUCTION "Mult" WITH "U"
1:Mult r0,r0,r4 ;MULTIPLY VALUES OF TOS REGISTER AND SOS REGISTER, AND PLACE RESULT INTO TOS REGISTER
<MICROPROGRAM FOR VIRTUAL MACHINE JUMP CODE WITH "U">

FIG. 40B

MICROPROGRAM FOR VIRTUAL MACHINE INSTRUCTION "Mult" WITH "D"
1:Mult r0,r0,r4 ;MULTIPLY VALUES OF TOS REGISTER AND SOS REGISTER, AND PLACE RESULT INTO TOS REGISTER
<MICROPROGRAM FOR VIRTUAL MACHINE JUMP CODE WITH "D">

FIG. 41A

MICROPROGRAM FOR VIRTUAL MACHINE JUMP CODE WITH "U"		
1:Load	r1,[r2]	;READ VIRTUAL MACHINE INSTRUCTION (JUMP ADDRESS) INDICATED BY VIRTUAL MACHINE PC INTO REGISTER #1
2:Inc	r2	;INCREMENT VALUE OF VIRTUAL MACHINE PC BY ONE
3:Jmp	r1	;JUMP UNCONDITIONALLY TO LOCATION INDICATED BY REGISTER #1

FIG. 41B

MICROPROGRAM FOR VIRTUAL MACHINE JUMP CODE WITH "D"		
1:Load	r1,[r2]	;READ VIRTUAL MACHINE INSTRUCTION (JUMP ADDRESS) INDICATED BY VIRTUAL MACHINE PC INTO REGISTER #1
2:Load	r4,[r3]	;READ VALUE FROM STACK AND COPY IT INTO SOS REGISTER
3:Inc	r2	;INCREMENT VALUE OF VIRTUAL MACHINE PC BY ONE
4:Dec	r3	;DECREMENT VALUE OF VIRTUAL MACHINE SP BY ONE
5:Jmp	r1	;JUMP UNCONDITIONALLY TO LOCATION INDICATED BY REGISTER #1

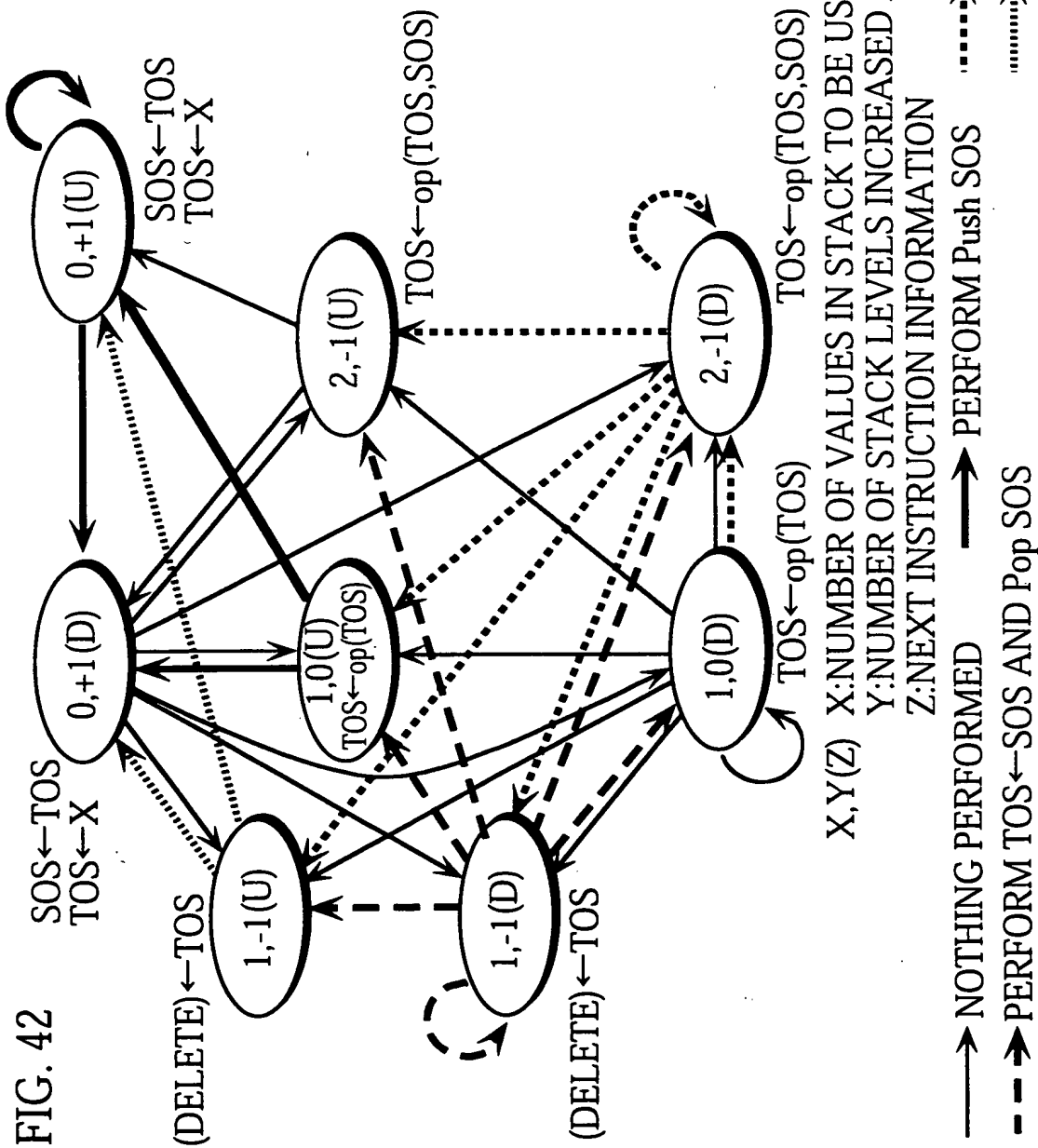


FIG. 43

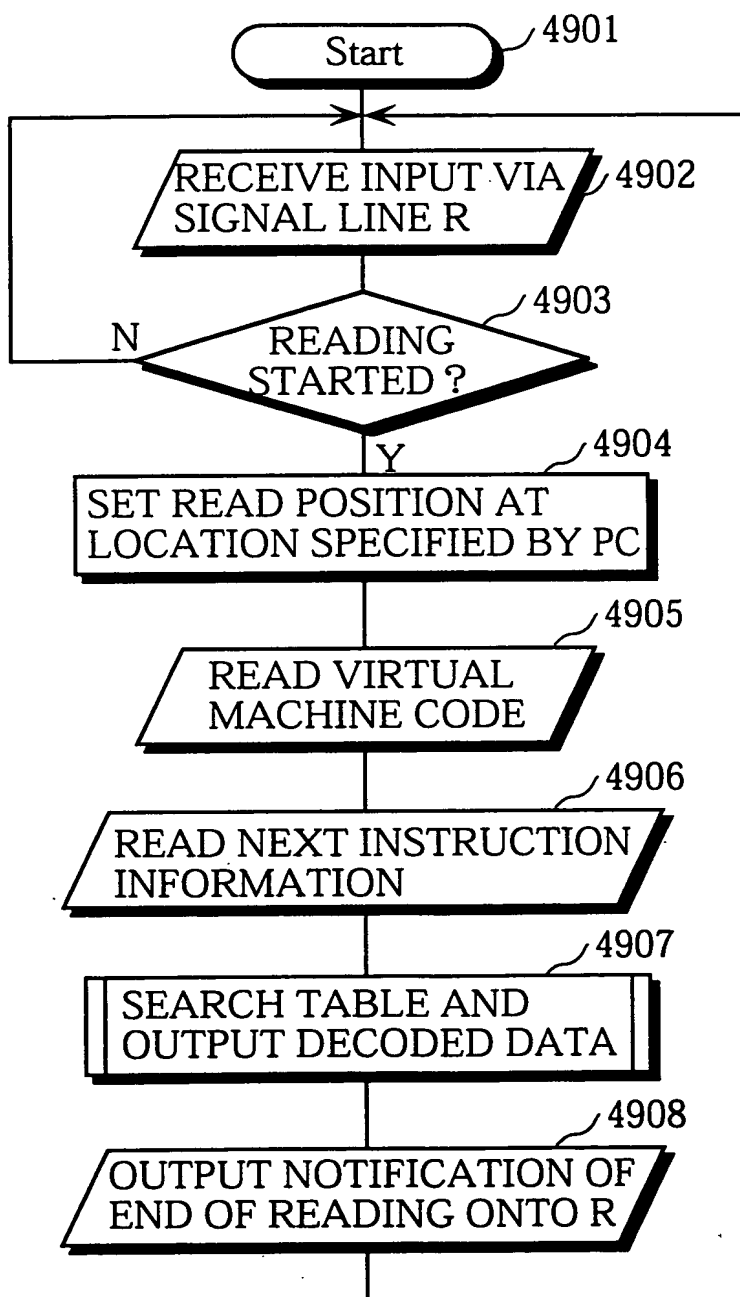


FIG. 44

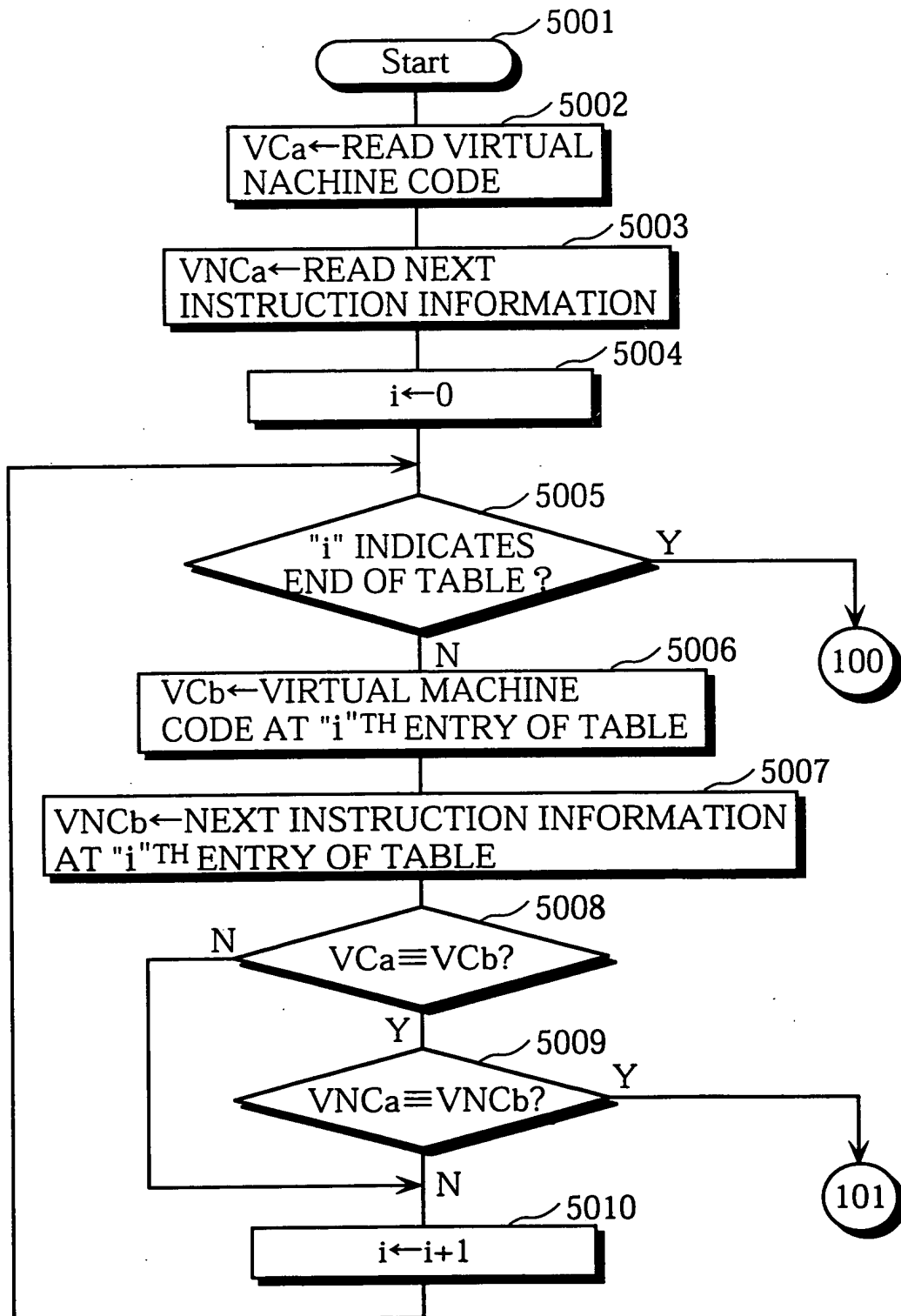


FIG. 45

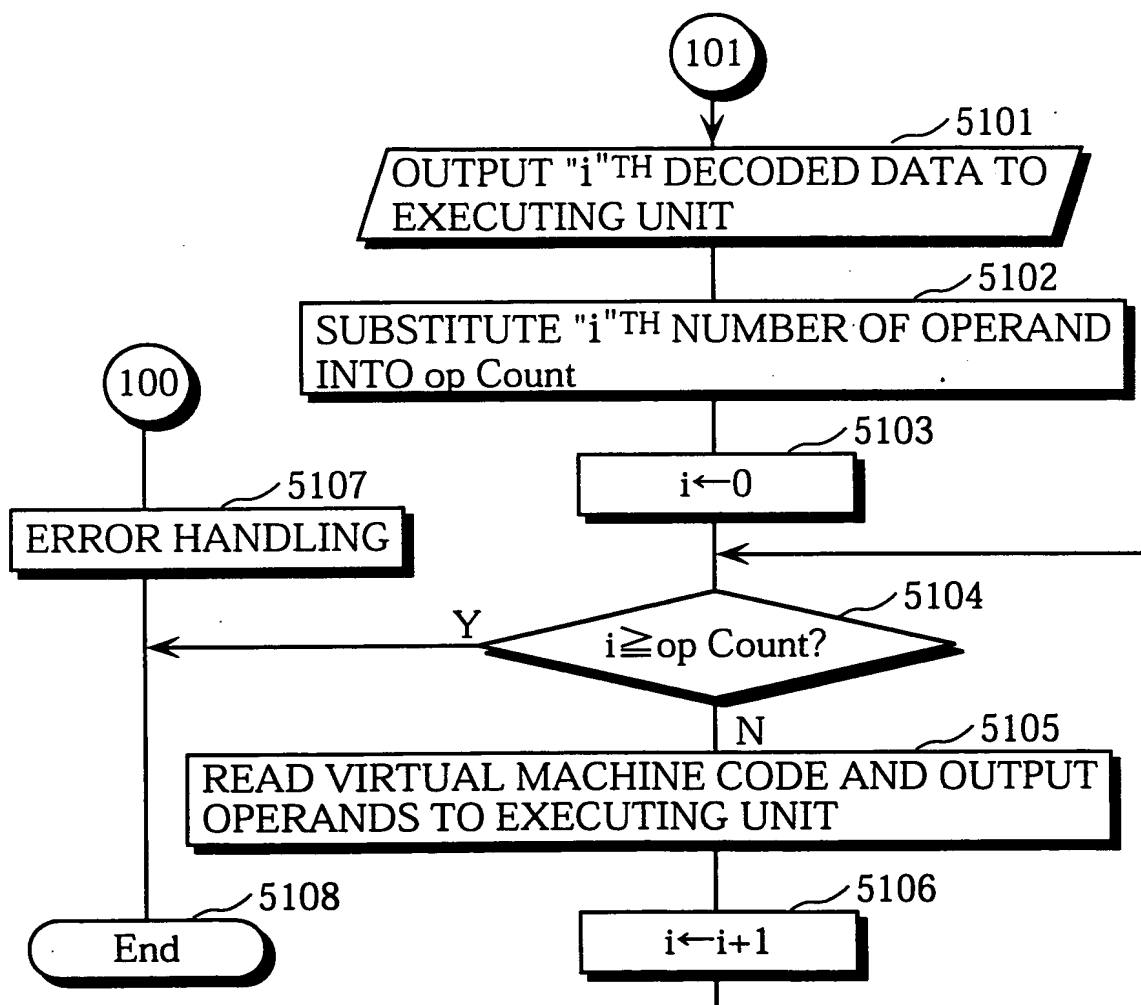




FIG. 46

1:<JUMP ADDRESS OF CODE TO PERFORM Push WITH "U">
2:OPERAND "2"
3:<JUMP ADDRESS OF CODE TO PERFORM Push WITH "U">
4:OPERAND "3"
5:<JUMP ADDRESS OF CODE TO PERFORM Push WITH "D">
6:OPERAND "4"
7:<JUMP ADDRESS OF CODE TO PERFORM Add WITH "D">
8:<JUMP ADDRESS OF CODE TO PERFORM Mult WITH "D">
9:<JUMP ADDRESS OF CODE TO PERFORM Pop WITH "U">
10:OPERAND "0"

FIG. 47A

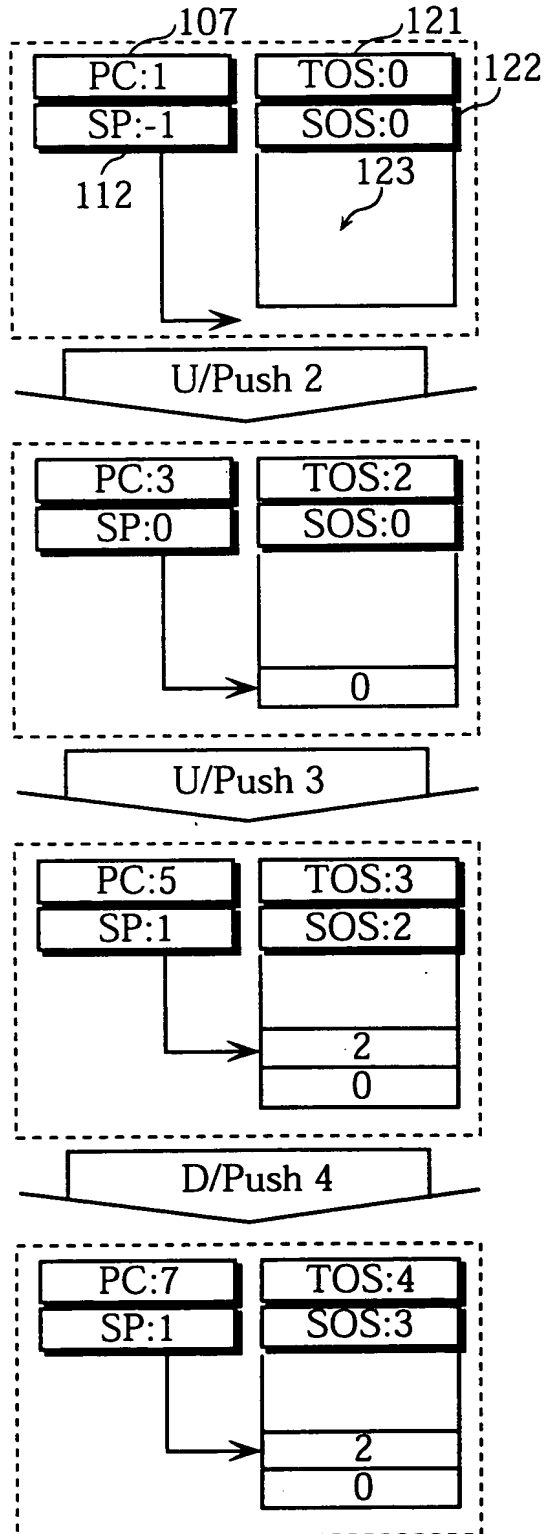


FIG. 47B

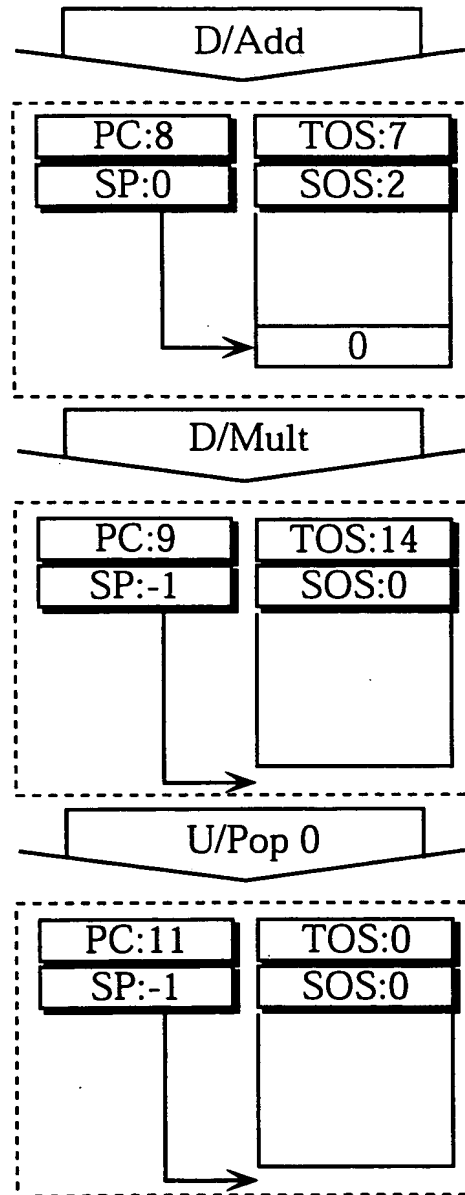




FIG. 48

CLOCK	1	2	3	4	5	6	7	8	9	10	11
Load r1,[r2]	IF	RF	ALU	MEM	WB						
Load r4,[r3]		IF	RF	ALU	MEM	WB					
Inc r2			IF	RF	ALU	MEM	WB				
Dec r3				IF	RF	ALU	MEM	WB			
Jmp r1					IF	RF	ALU	MEM	WB		
						IF	x				
Mult r0,r0,r4							IF	RF	ALU	MEM	WB

FIG. 49

CLOCK	1	2	3	4	5	6	7	8	9
Load r1,[r2]	IF	RF	ALU	MEM	WB				
Load r4,[r3]	IF	RF	ALU	MEM	WB				
Inc r2		IF	RF	ALU	MEM	WB			
Dec r3		IF	RF	ALU	MEM	WB			
Jmp r1			IF	RF	ALU	MEM	WB		
			IF	RF	x				
				IF	x				
				IF	x				
Mult r0,r0,r4					IF	RF	ALU	MEM	WB



FIG. 50

CLOCK	1	2	3	4	5	6	7	8	9	10	11
Load r1,[r2]	IF	RF	ALU	MEM	WB						
Load r4,[r3]		IF	RF	ALU	MEM	WB					
Inc r2			IF	RF	ALU	MEM	WB				
Dec r3				IF	RF	ALU	MEM	WB			
Jmp r1					IF	RF	ALU	MEM	WB		
						IF	x				
Mult r0,r0,r4							IF	RF	ALU	MEM	WB

FIG. 51

CLOCK	1	2	3	4	5	6	7	8	9	10
Load r1,[r2]	IF	RF	ALU	MEM	WB					
Load r4,[r3]	IF	RF	ALU	MEM	WB					
Inc r2		IF	RF	ALU	MEM	WB				
Dec r3		IF	RF	ALU	MEM	WB				
Jmp r1			IF	RF	•	ALU	MEM	WB		
			IF	RF	ALU	x				
				IF	RF	x				
				IF	RF	x				
					IF	x				
					IF	x				
Mult r0,r0,r4						IF	RF	ALU	MEM	WB

FIG. 52

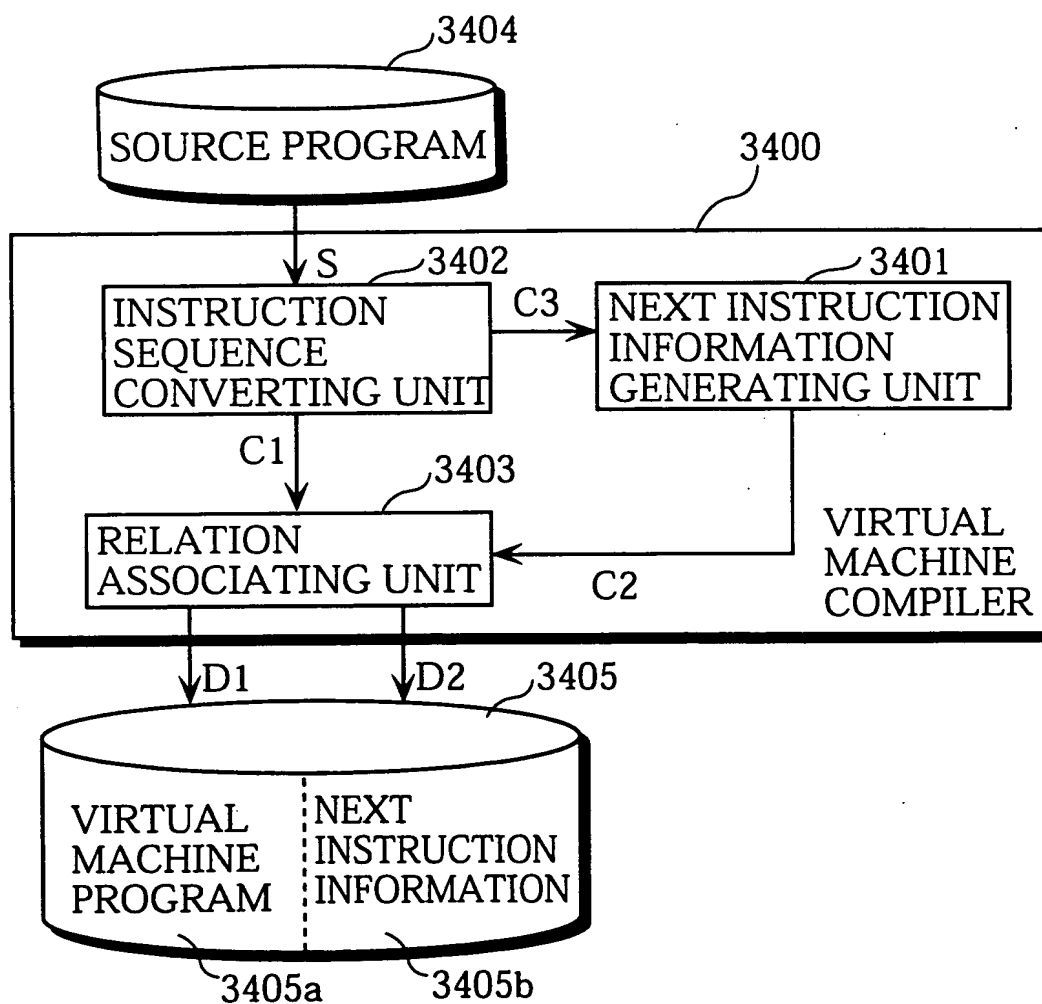


FIG. 53

DATA CONSTRUCTION OF

INSTRUCTION SEQUENCE: "x=(1+2)*(3+4)"

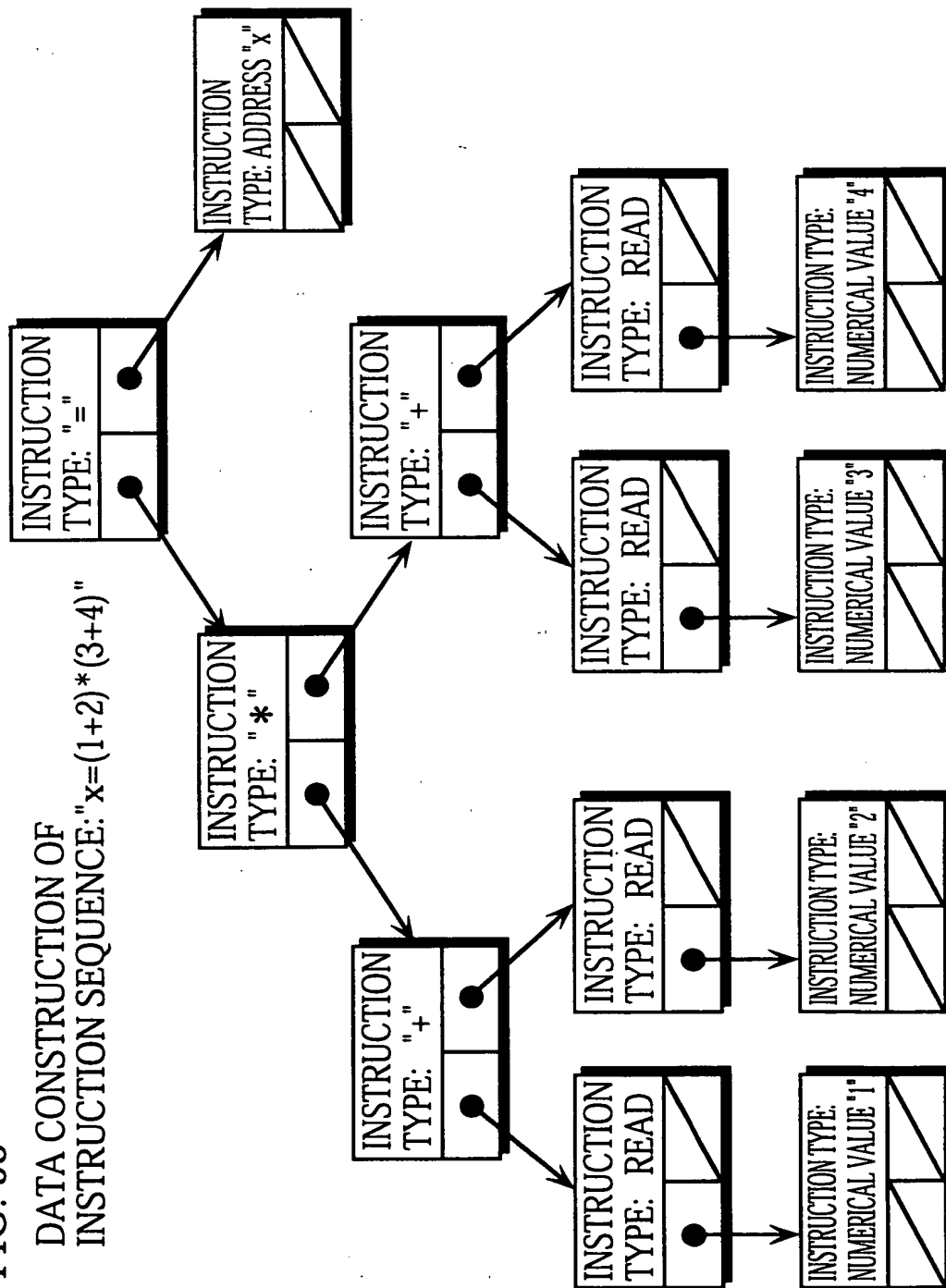


FIG. 54

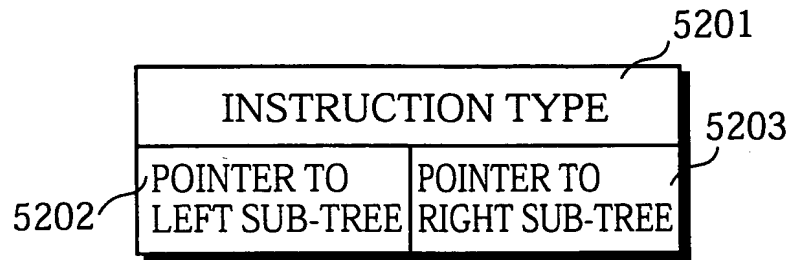


FIG. 55

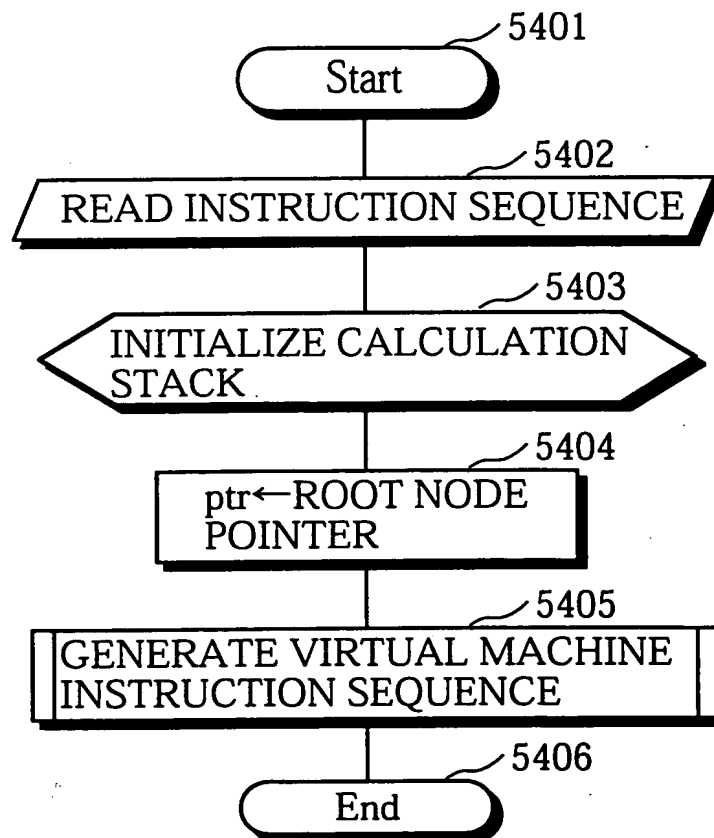
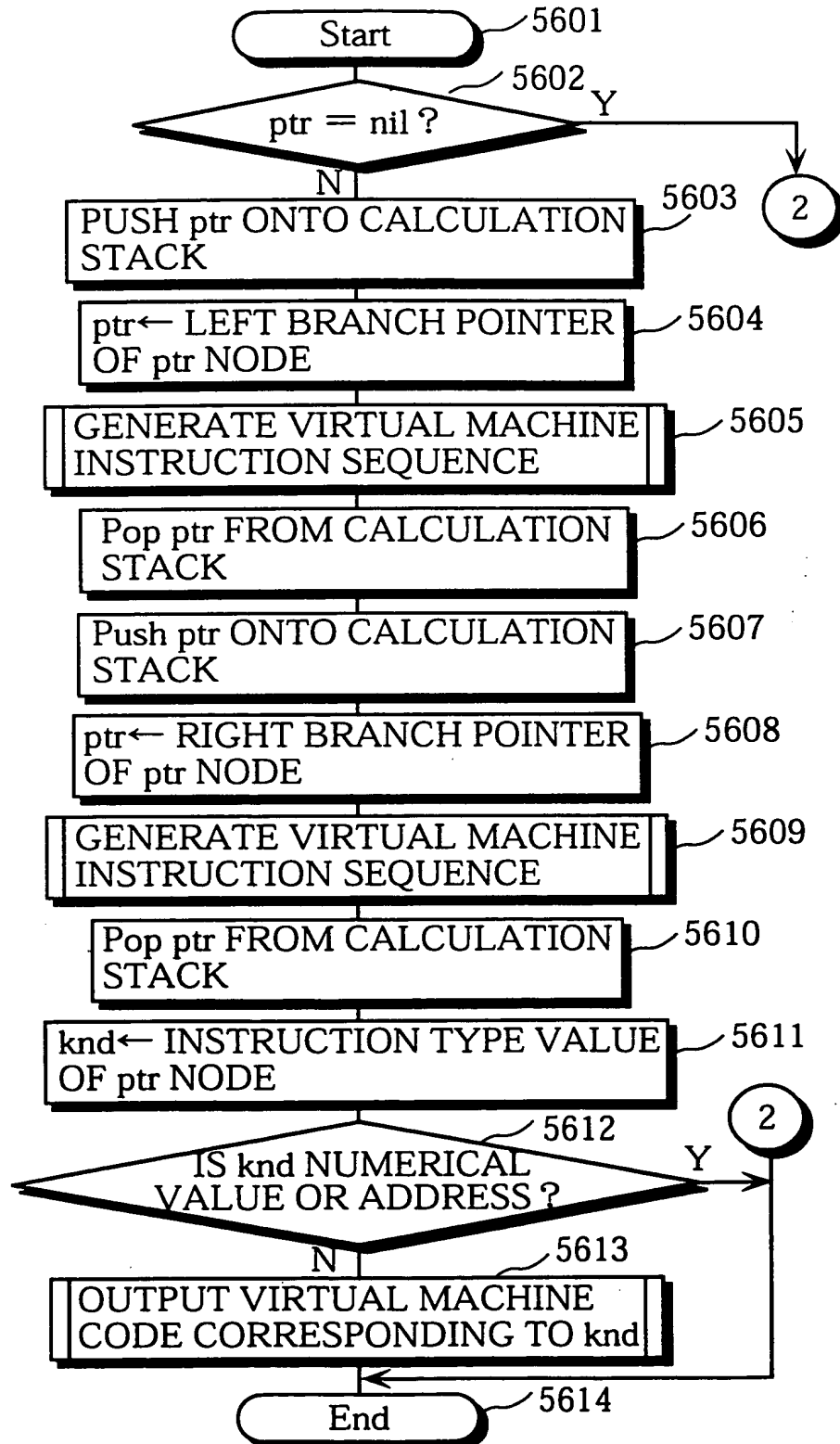


FIG. 56



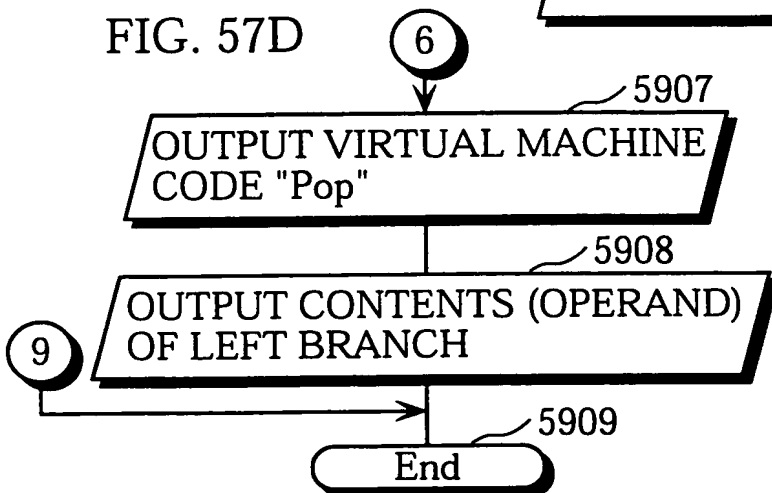
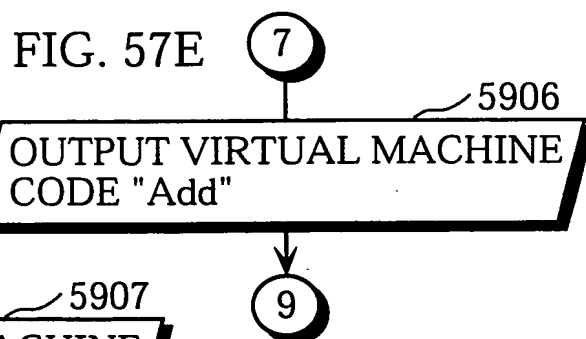
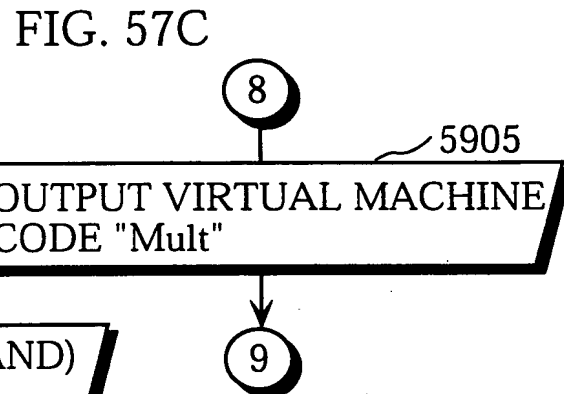
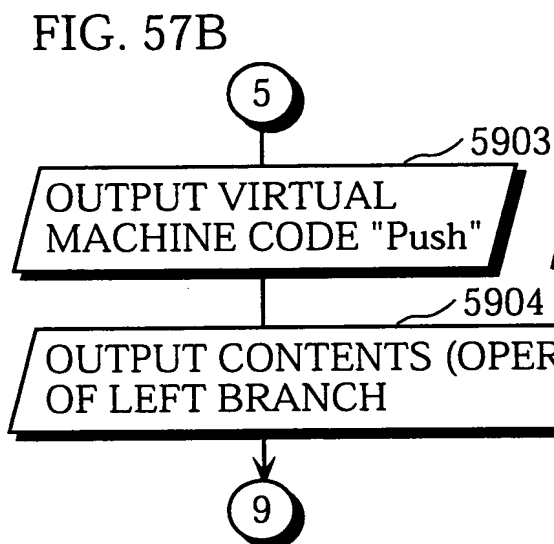
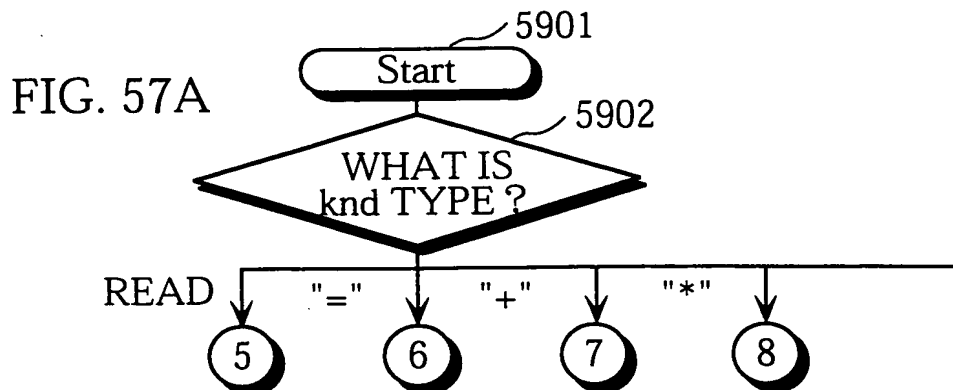


FIG. 58

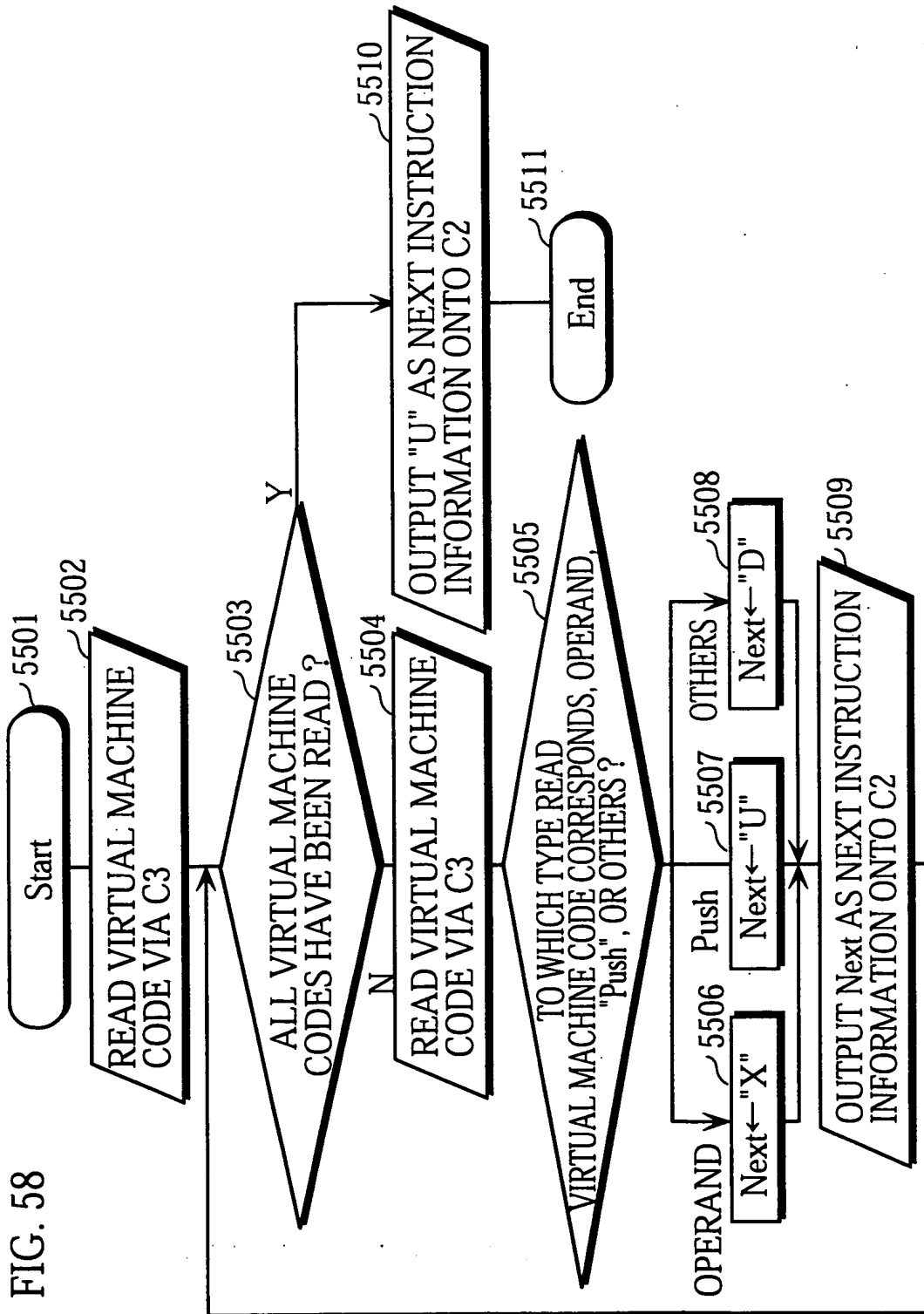


FIG. 59

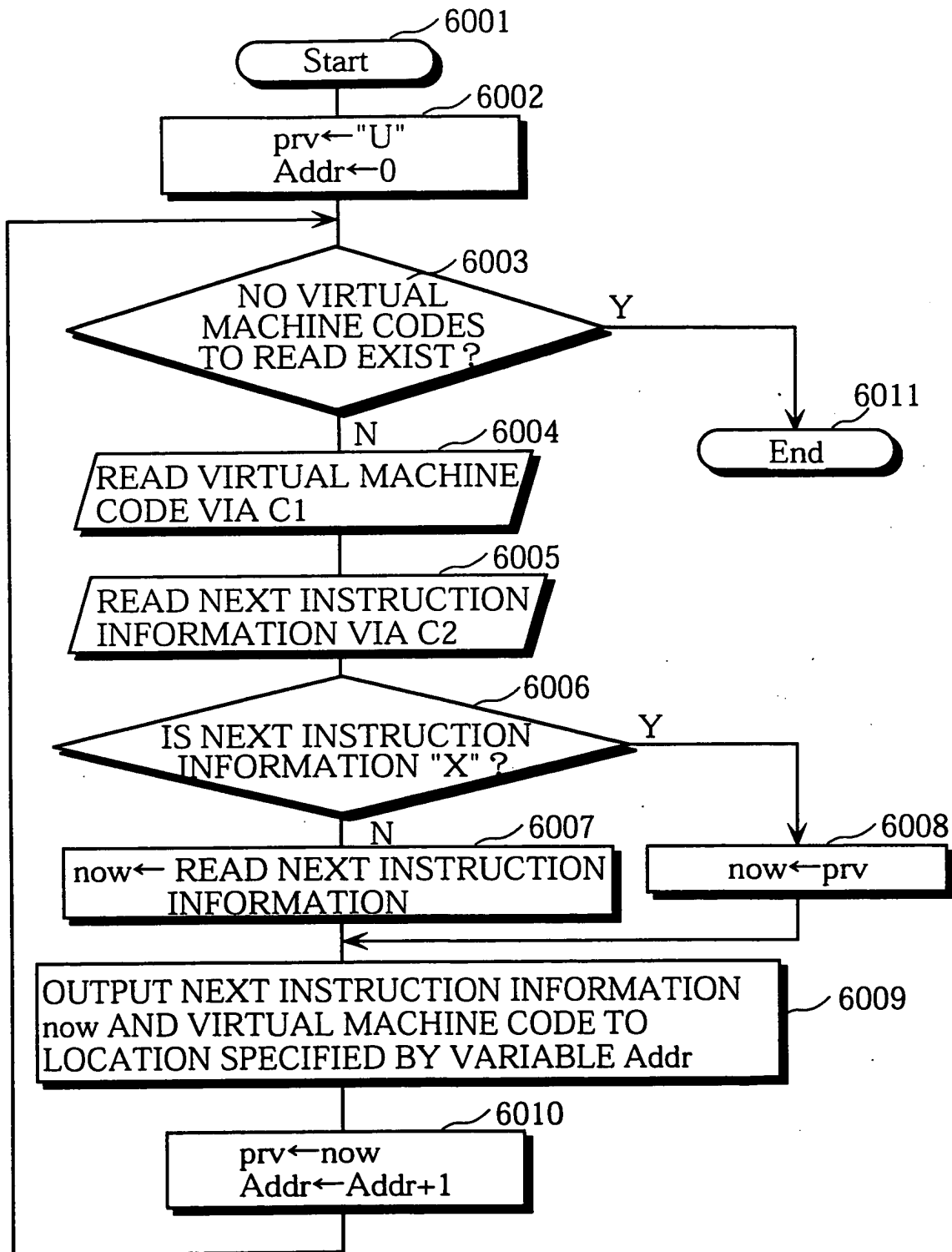


FIG. 60

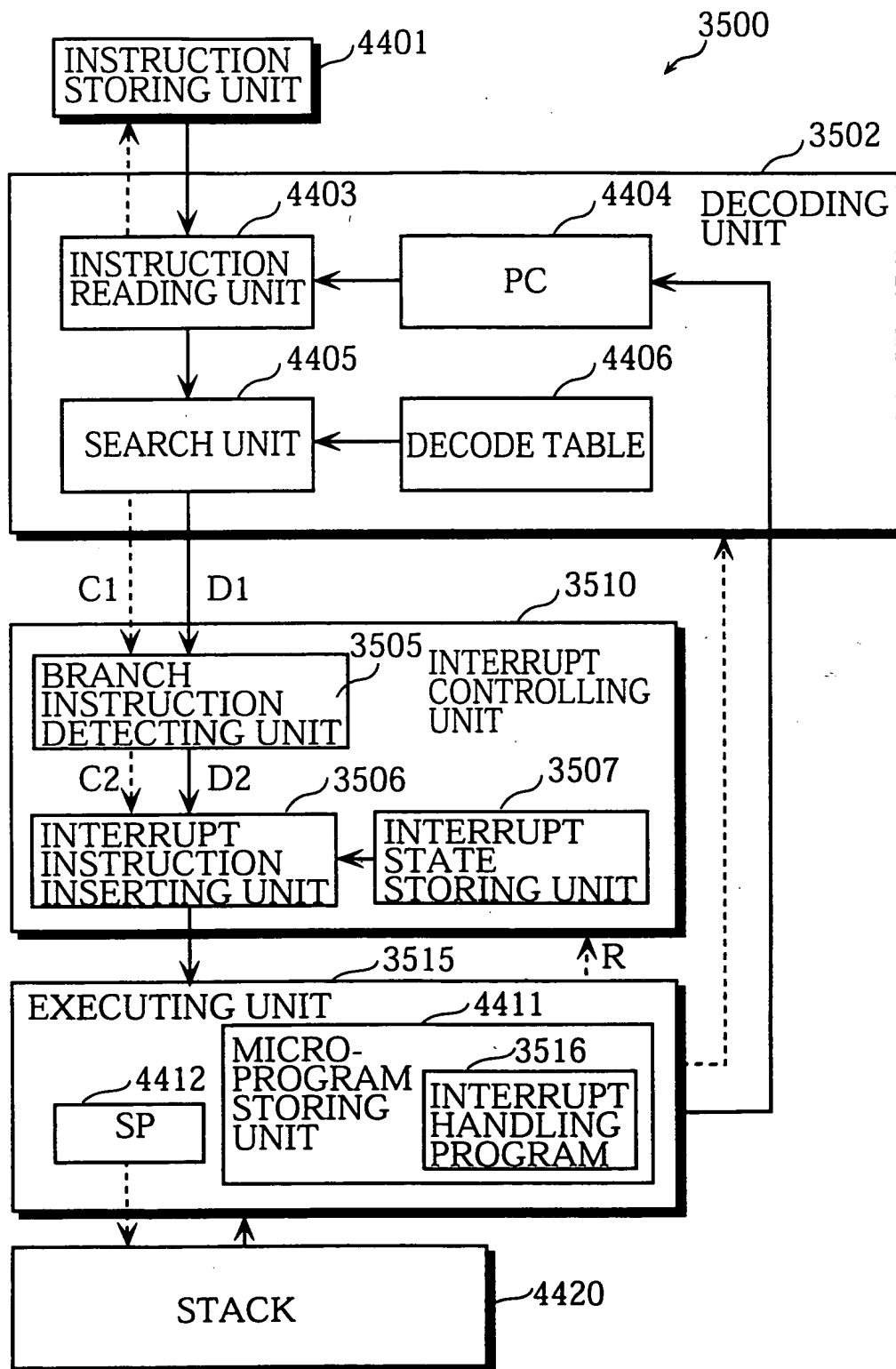


FIG. 61

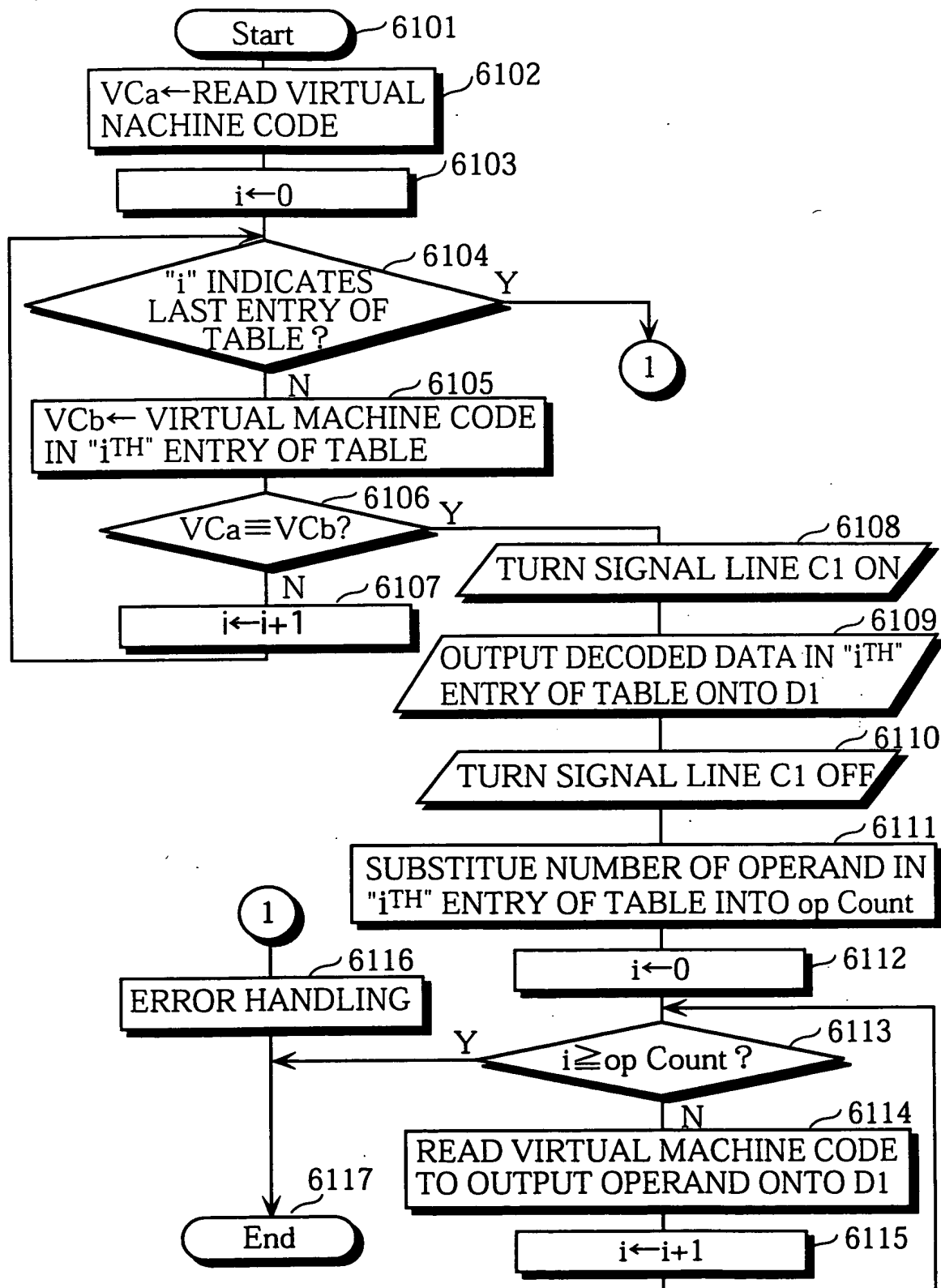


FIG. 62

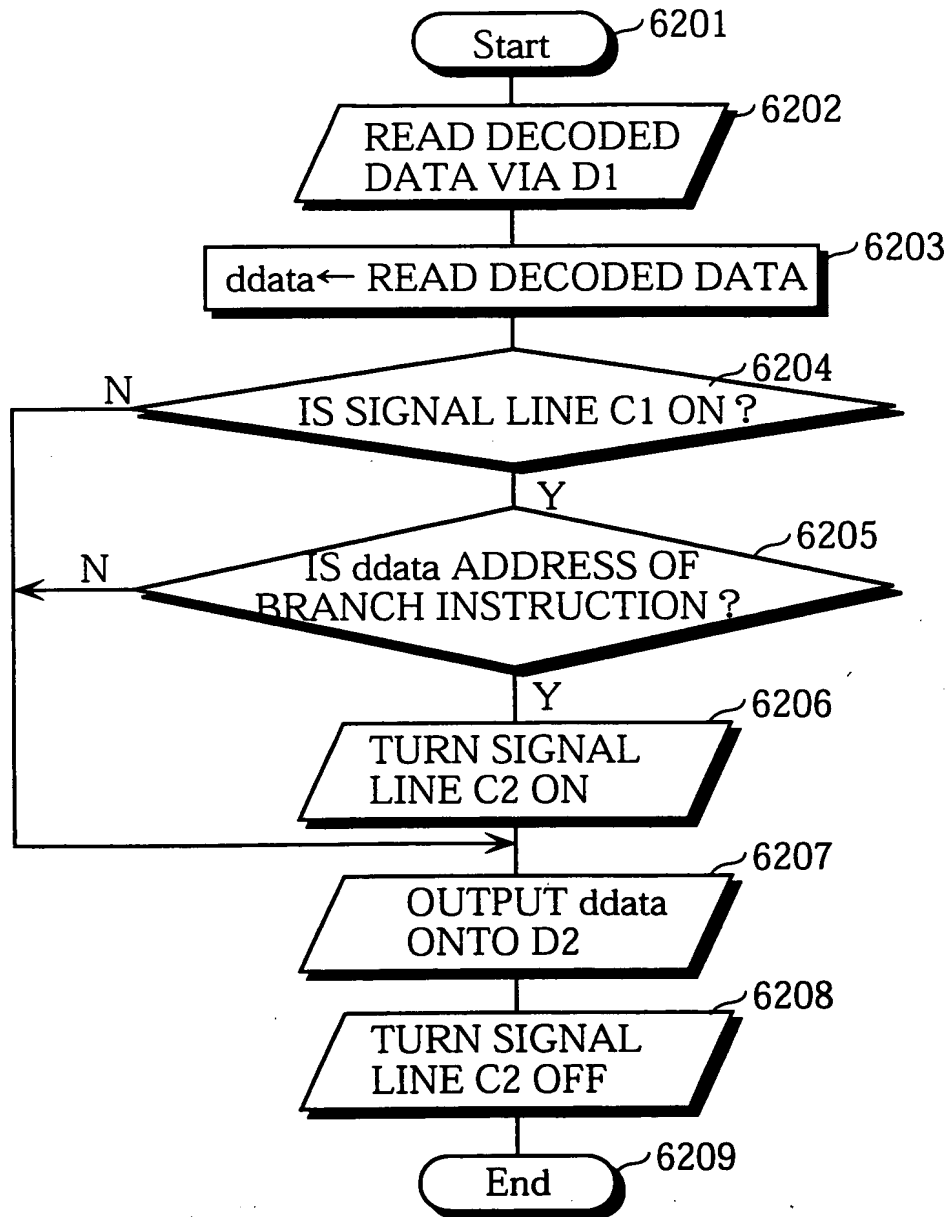


FIG. 63

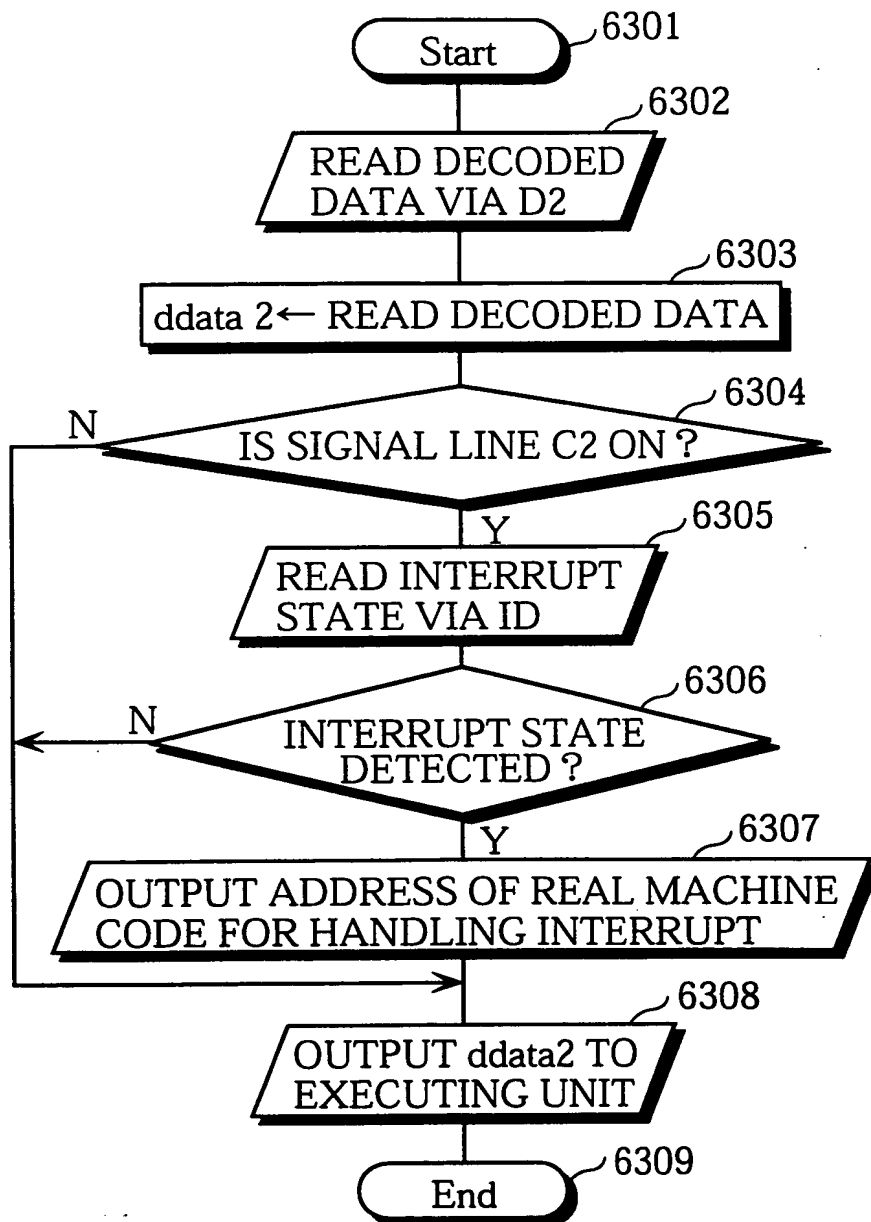


FIG. 64

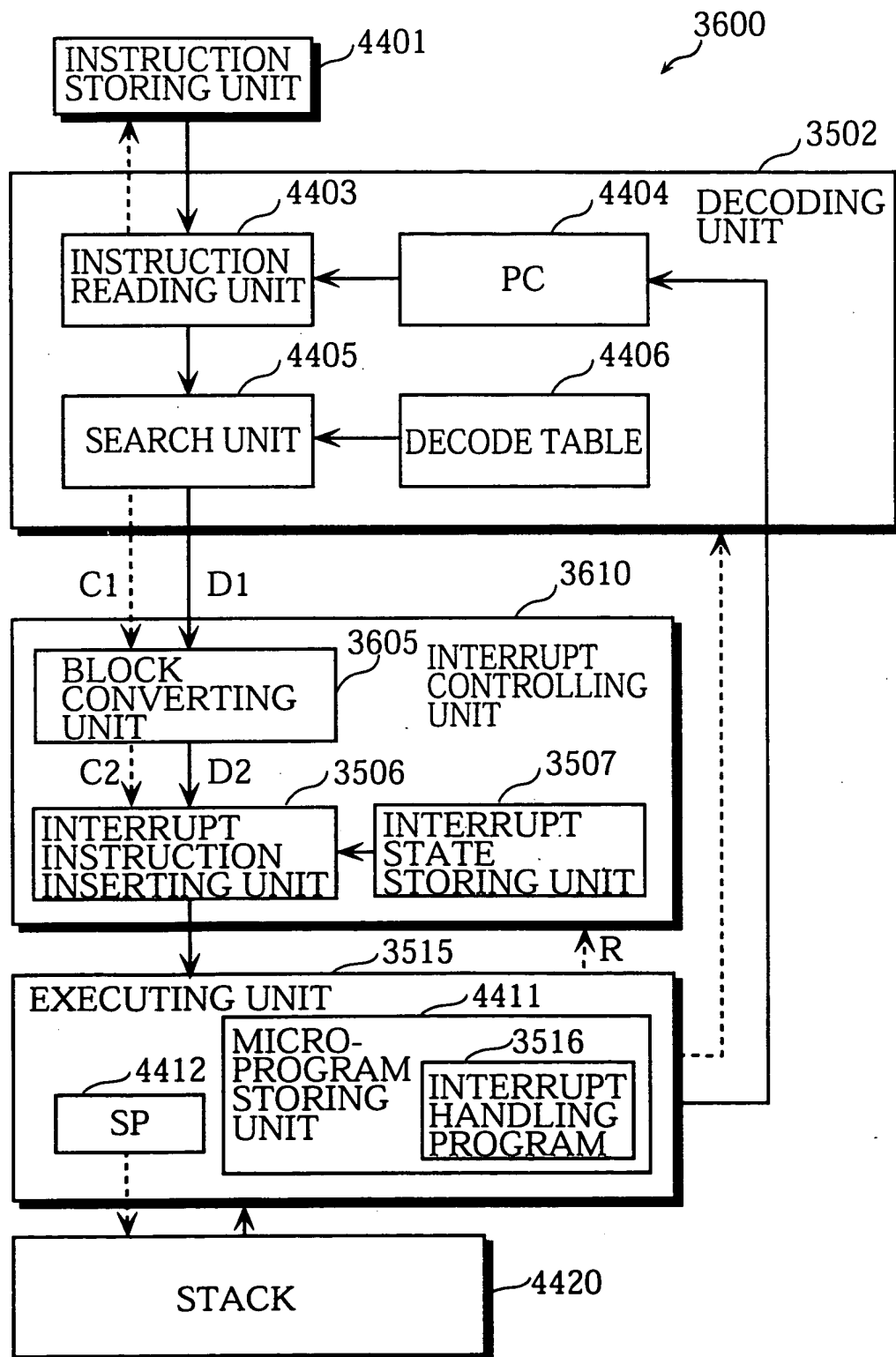


FIG. 65

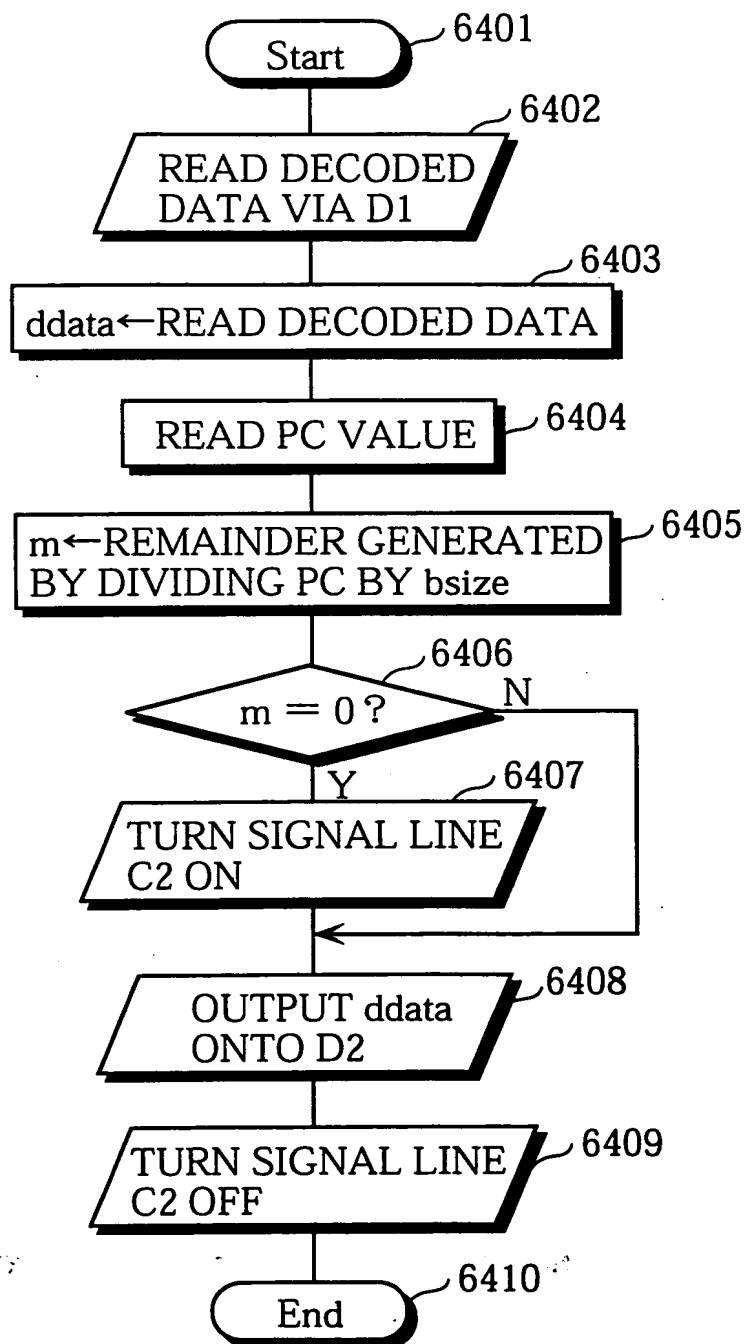


FIG. 66

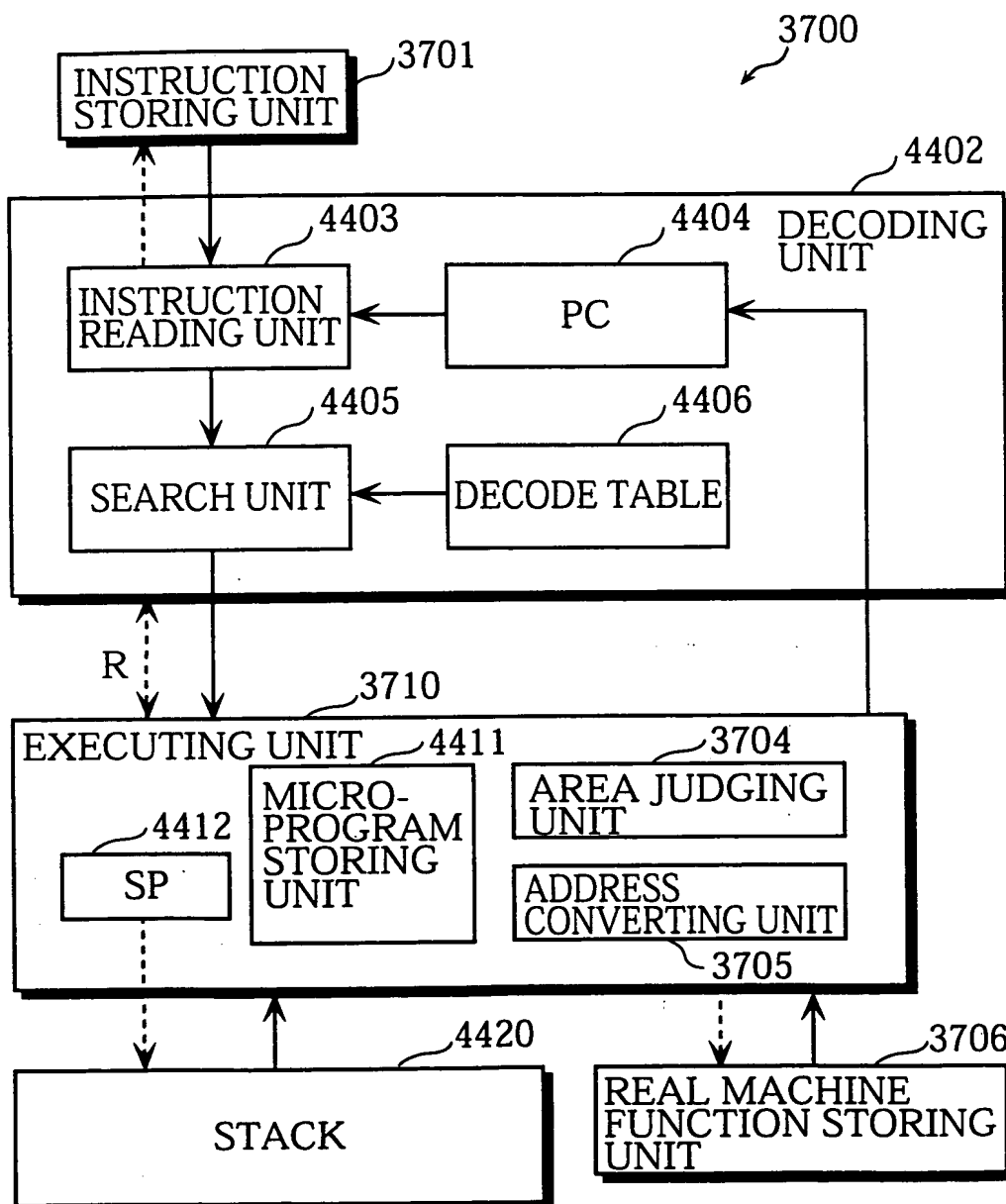




FIG. 67

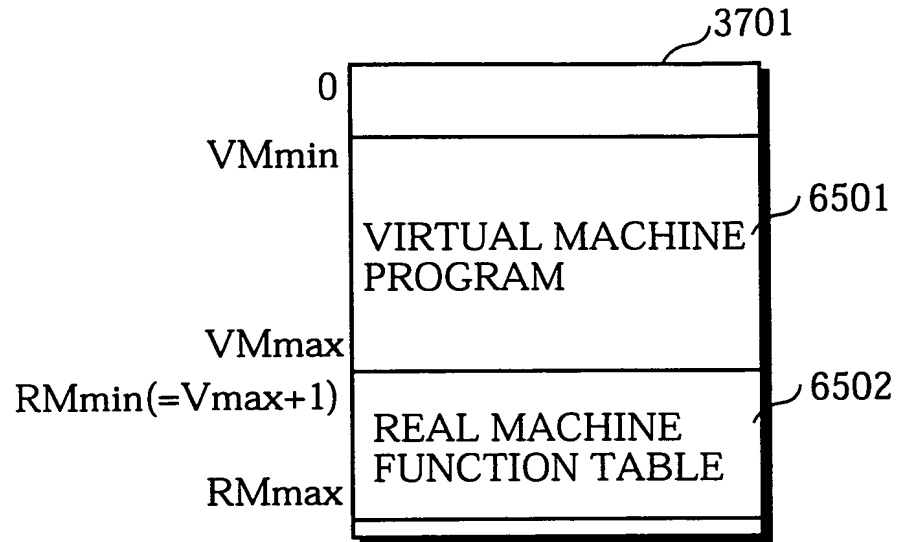


FIG. 68

(CORRESPONDING
ADDRESS)

RMmax	(RMmax-RMmin) th POINTER TO REAL MACHINE FUNCTION
RMmax-1	(RMmax-RMmin-1) th POINTER TO REAL MACHINE FUNCTION
RMmax-2	(RMmax-RMmin-2) th POINTER TO REAL MACHINE FUNCTION
	:
RMmin	0 th POINTER TO REAL MACHINE FUNCTION

FIG. 69

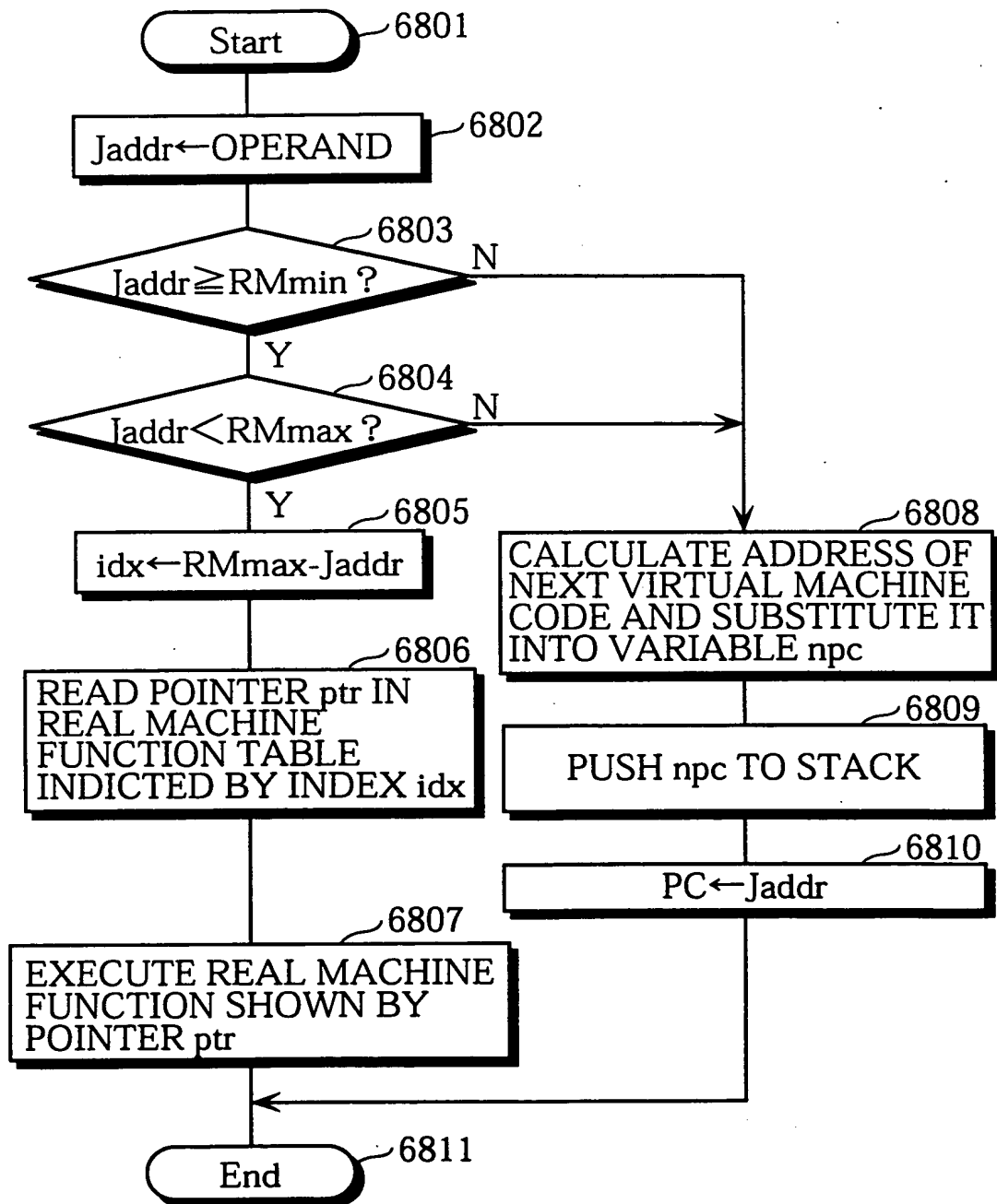


FIG. 70

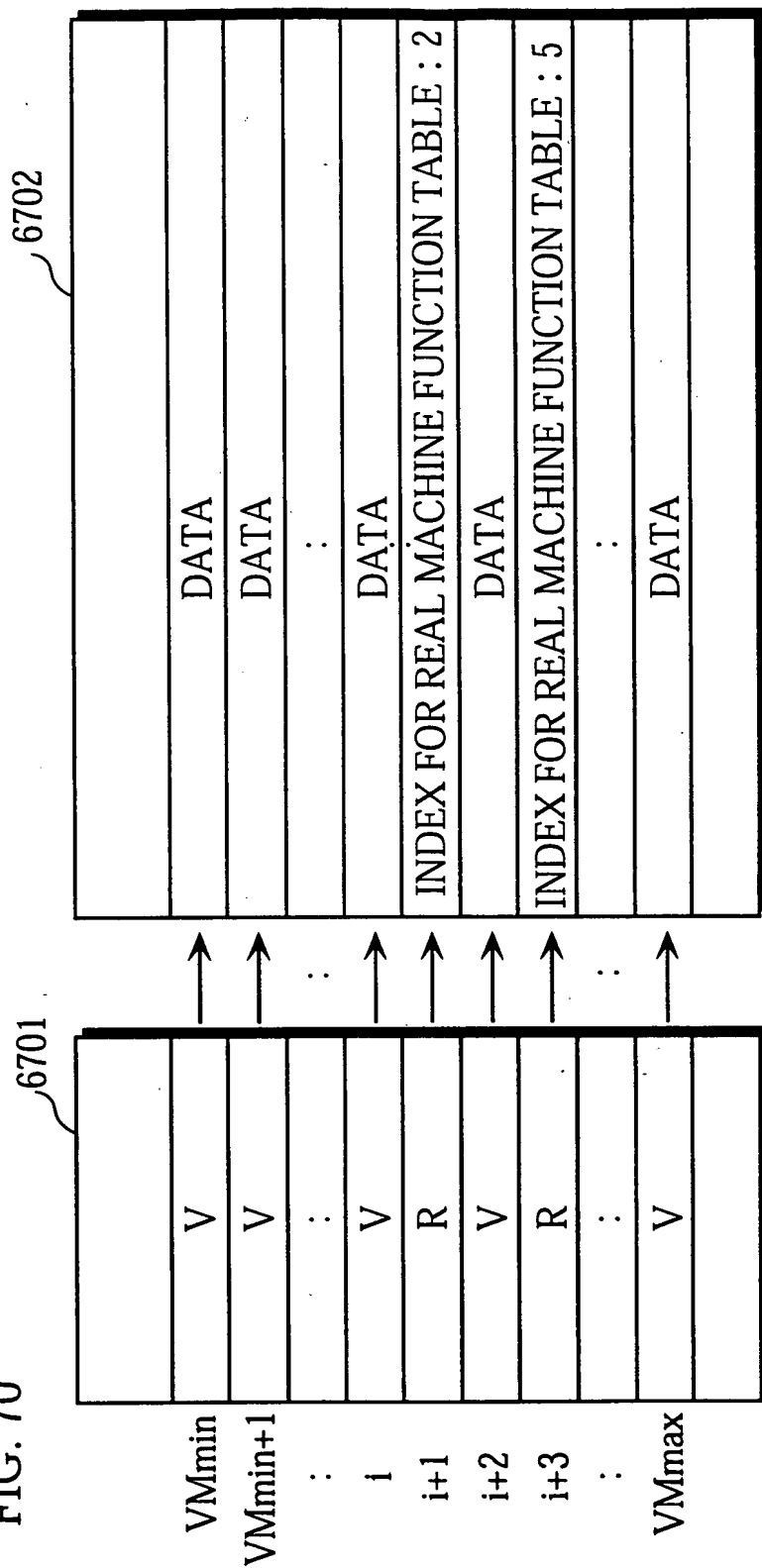


FIG. 71

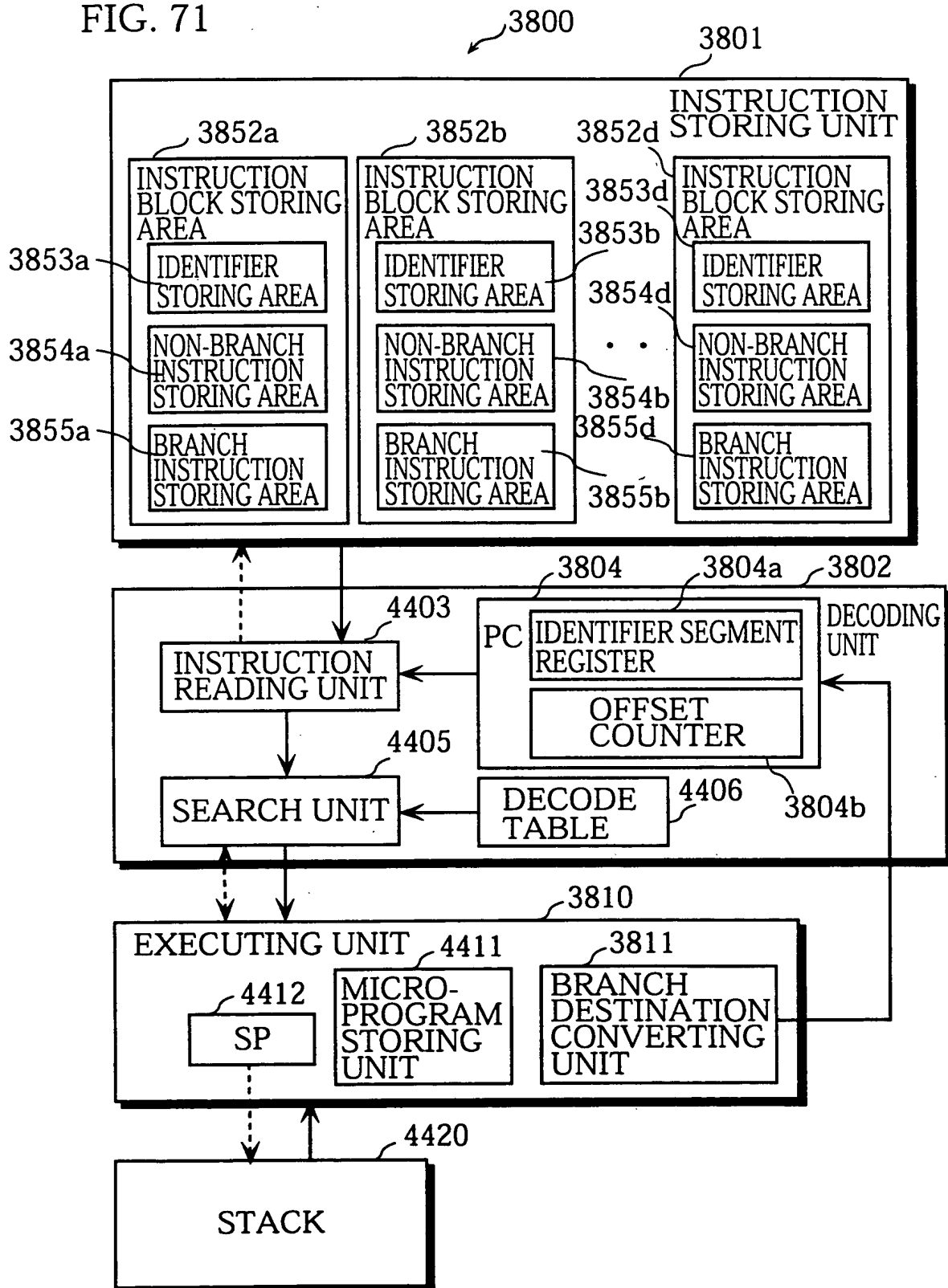


FIG. 72

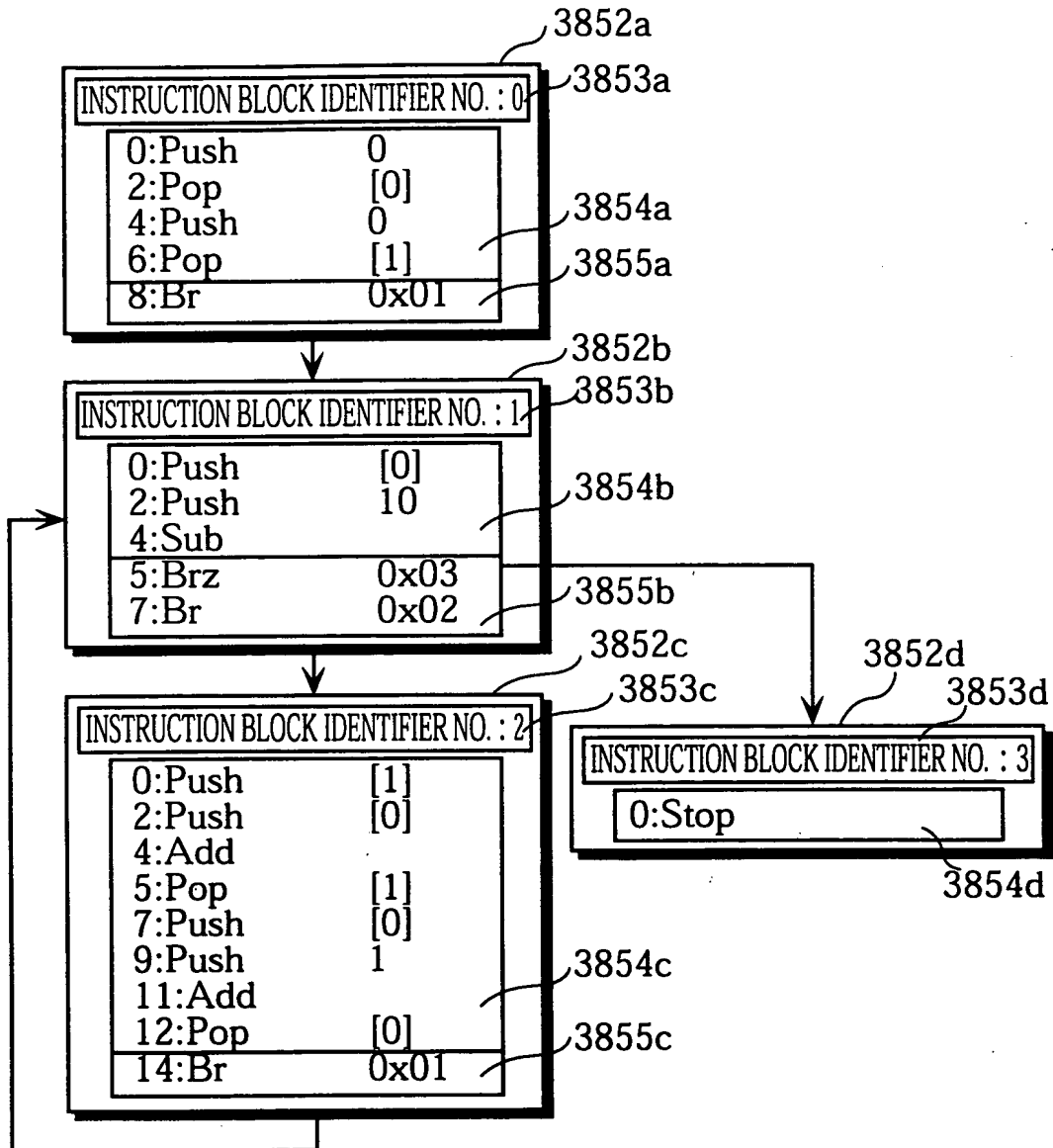




FIG. 73

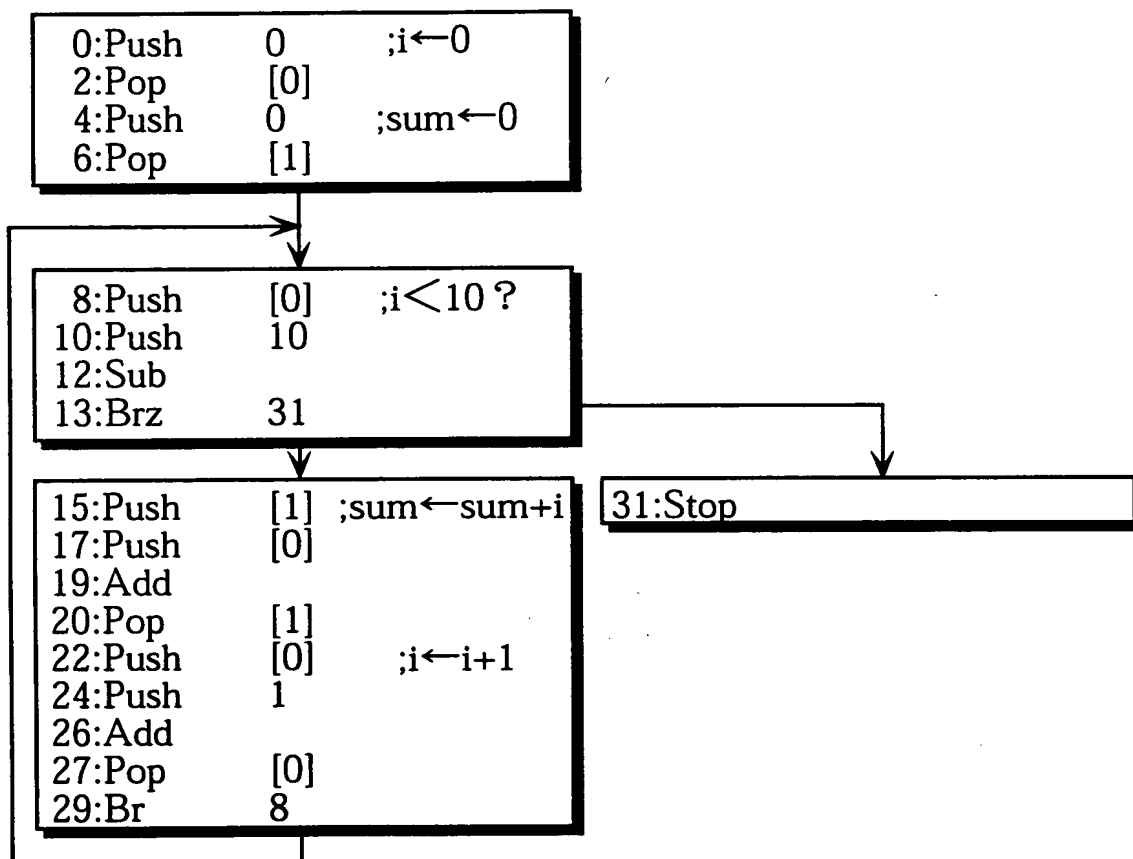




FIG. 74

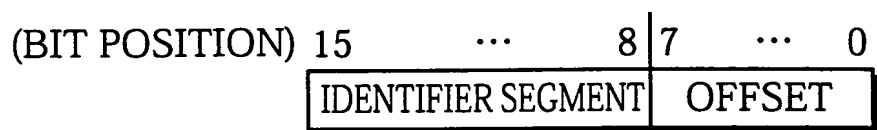


FIG. 75

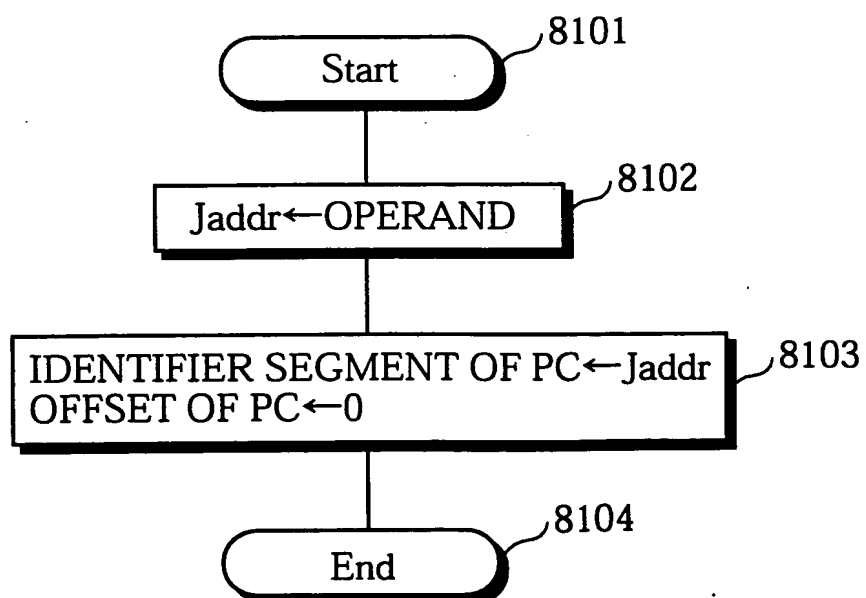




FIG. 76

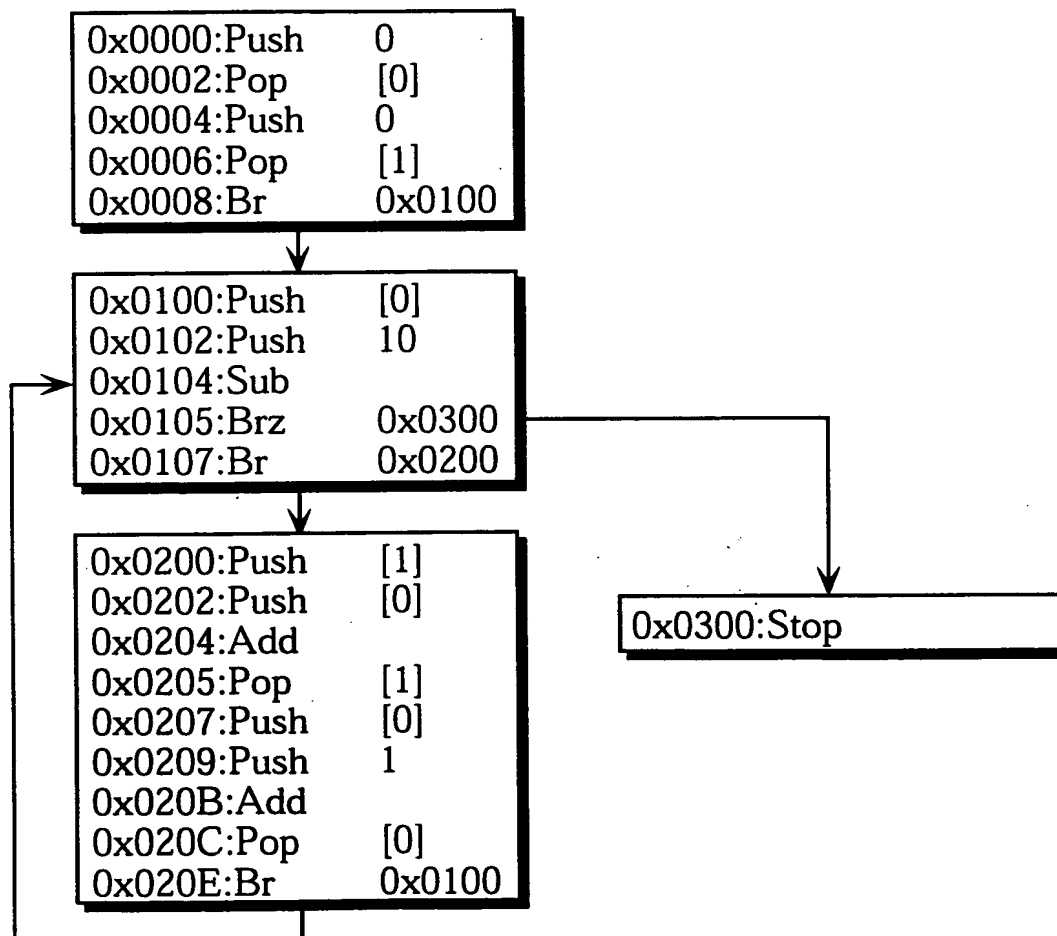


FIG. 77

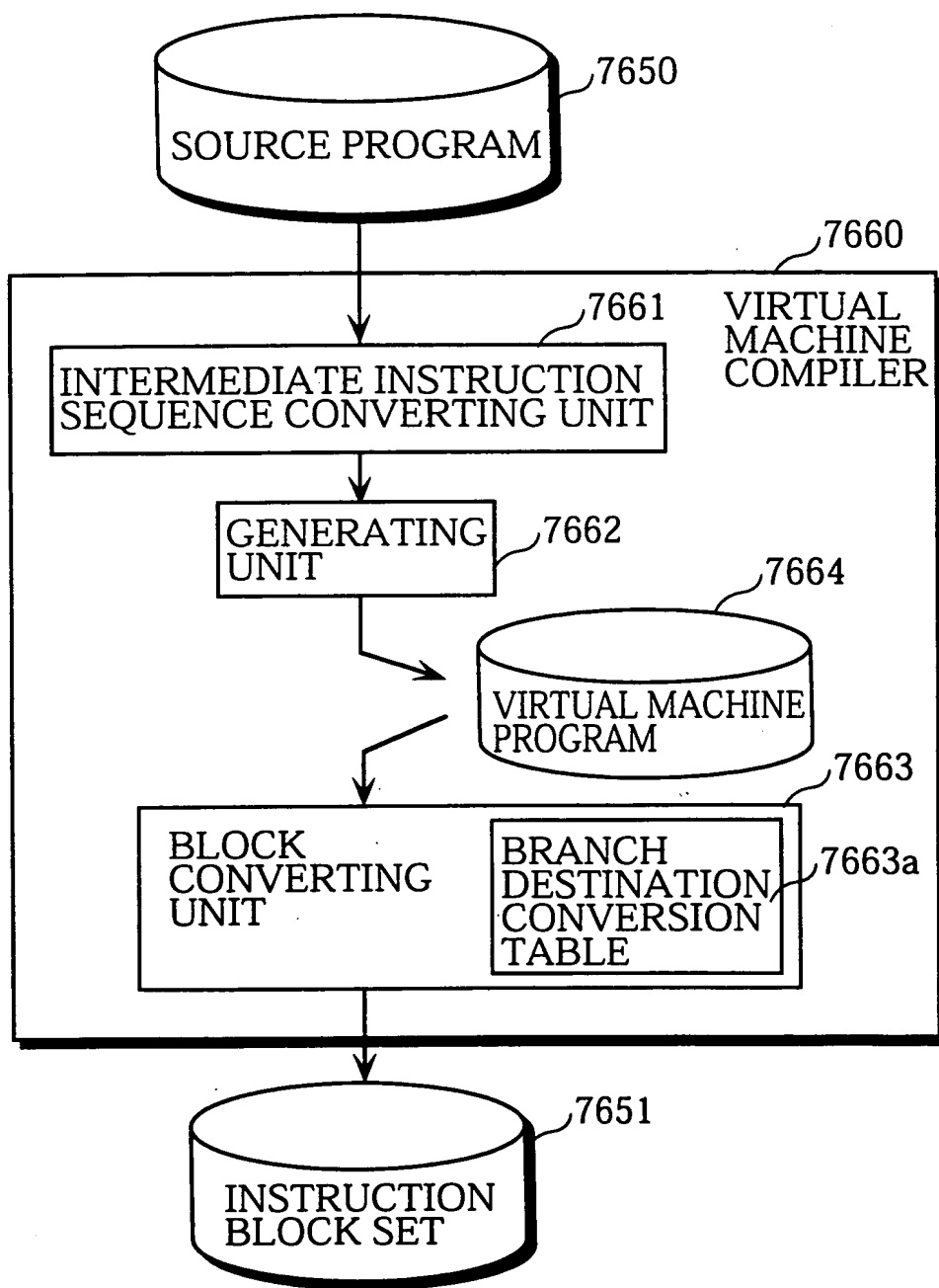




FIG. 78

7663a

0	CODE POSITION :	REGISTRATION FLAG :	REFERENCE POSITION OFFSET :	REFERENCE POSITION IDENTIFIER :
1	CODE POSITION :	REGISTRATION FLAG :	REFERENCE POSITION OFFSET :	REFERENCE POSITION IDENTIFIER :
2	CODE POSITION :	REGISTRATION FLAG :	REFERENCE POSITION OFFSET :	REFERENCE POSITION IDENTIFIER :
3	CODE POSITION :	REGISTRATION FLAG :	REFERENCE POSITION OFFSET :	REFERENCE POSITION IDENTIFIER :
4	CODE POSITION :	REGISTRATION FLAG :	REFERENCE POSITION OFFSET :	REFERENCE POSITION IDENTIFIER :

⋮

Rcount-1	CODE POSITION :	REGISTRATION FLAG :	REFERENCE POSITION OFFSET :	REFERENCE POSITION IDENTIFIER :
----------	-----------------	---------------------	-----------------------------	---------------------------------

FIG. 79

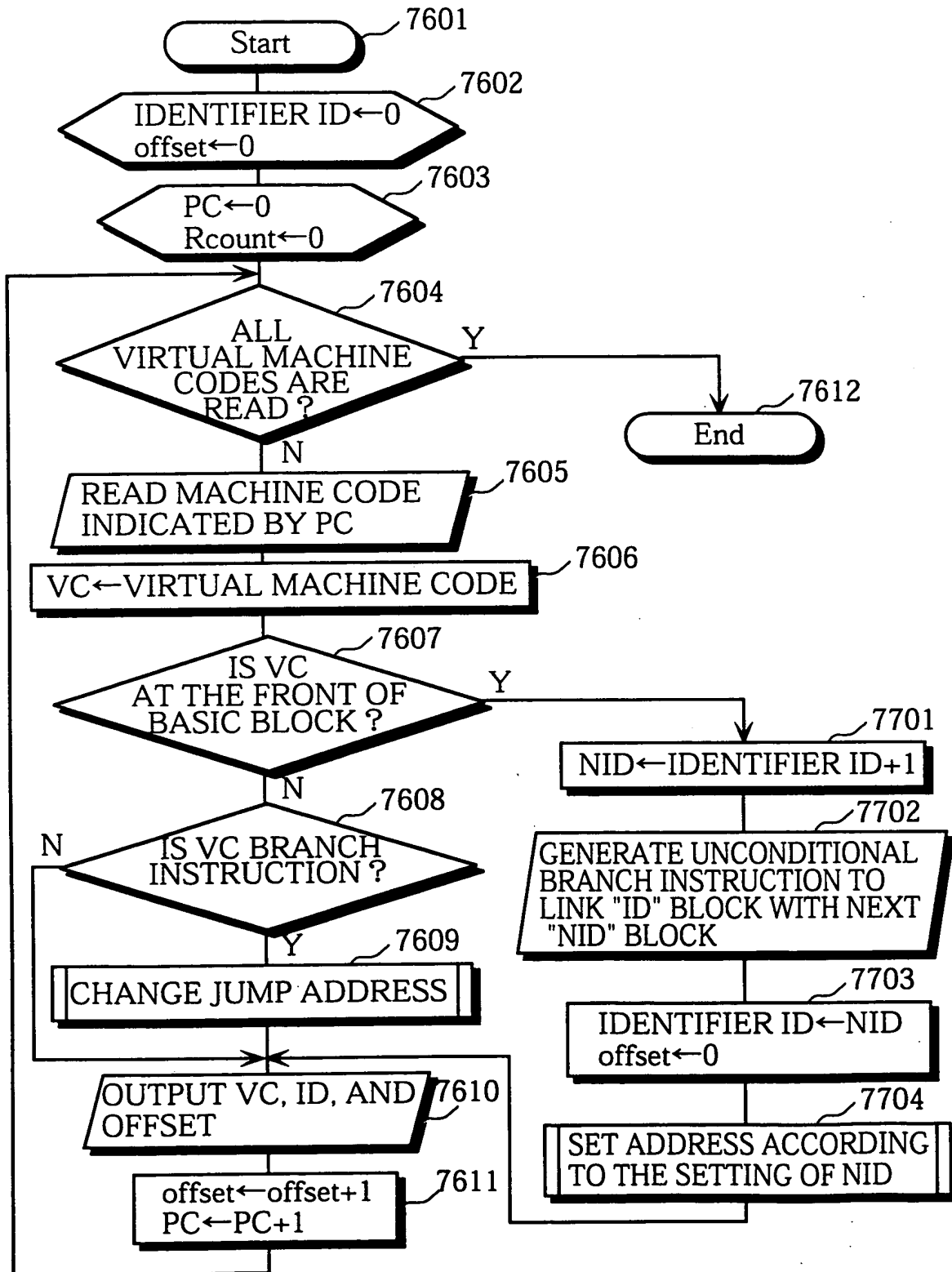


FIG. 80

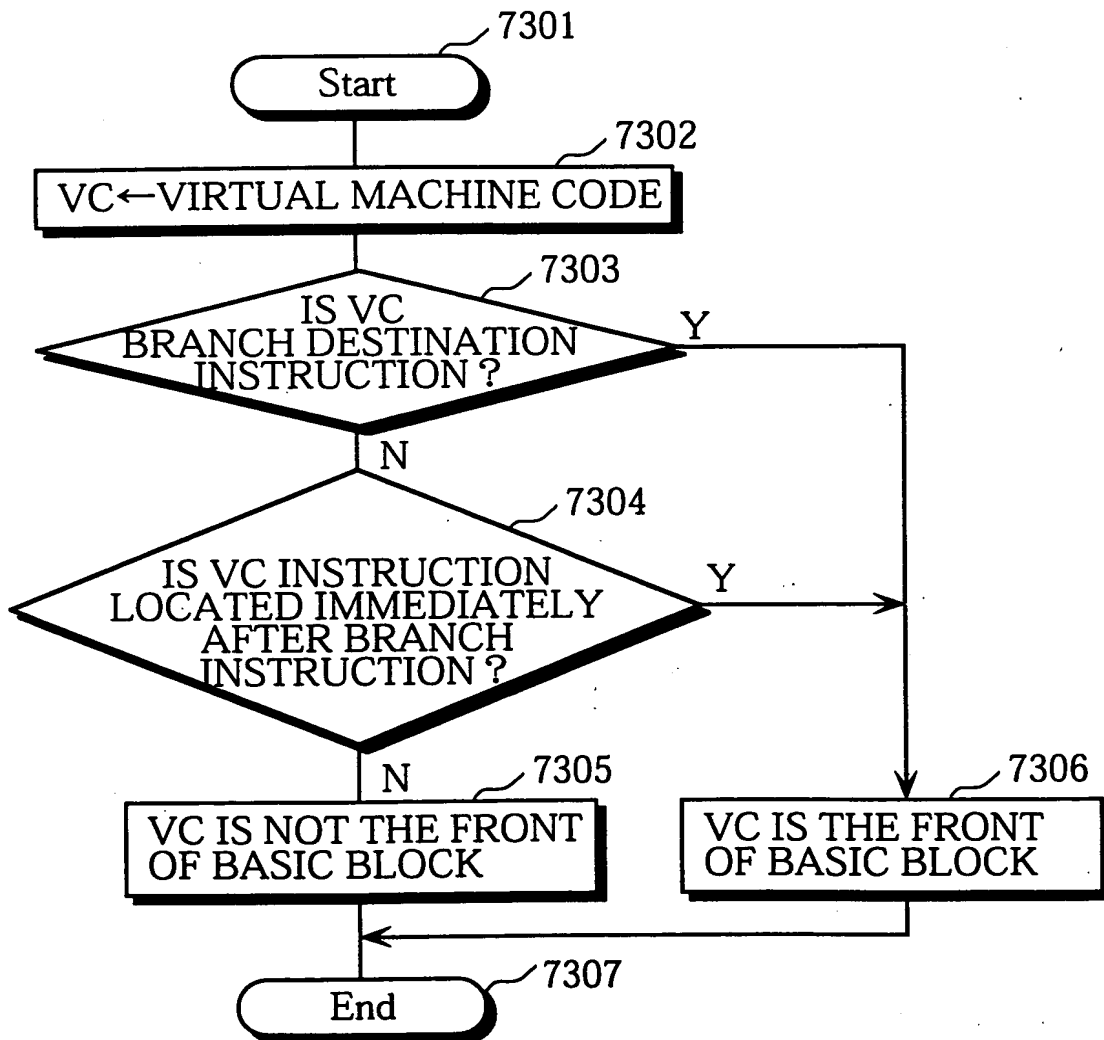


FIG. 81

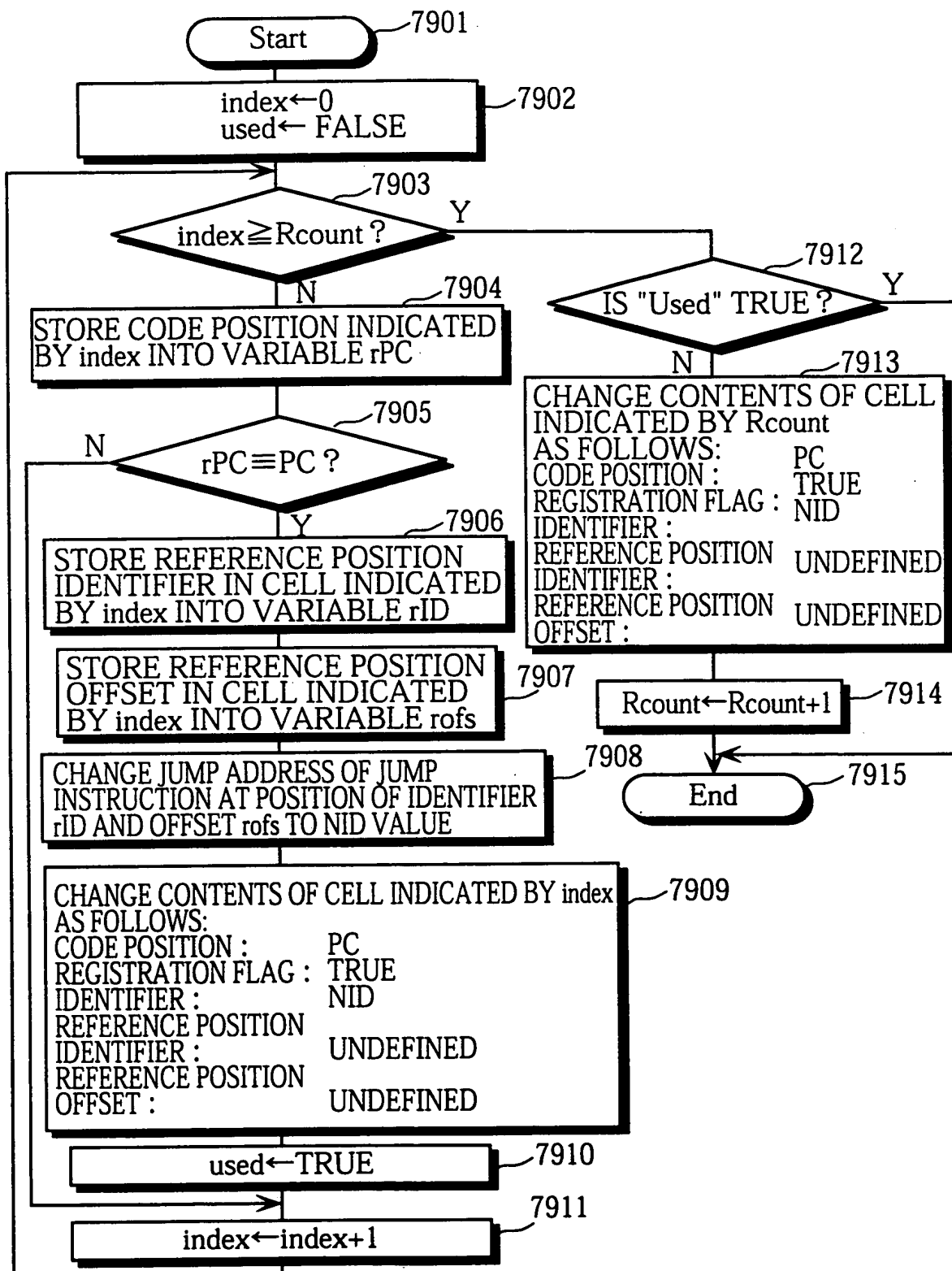


FIG. 82

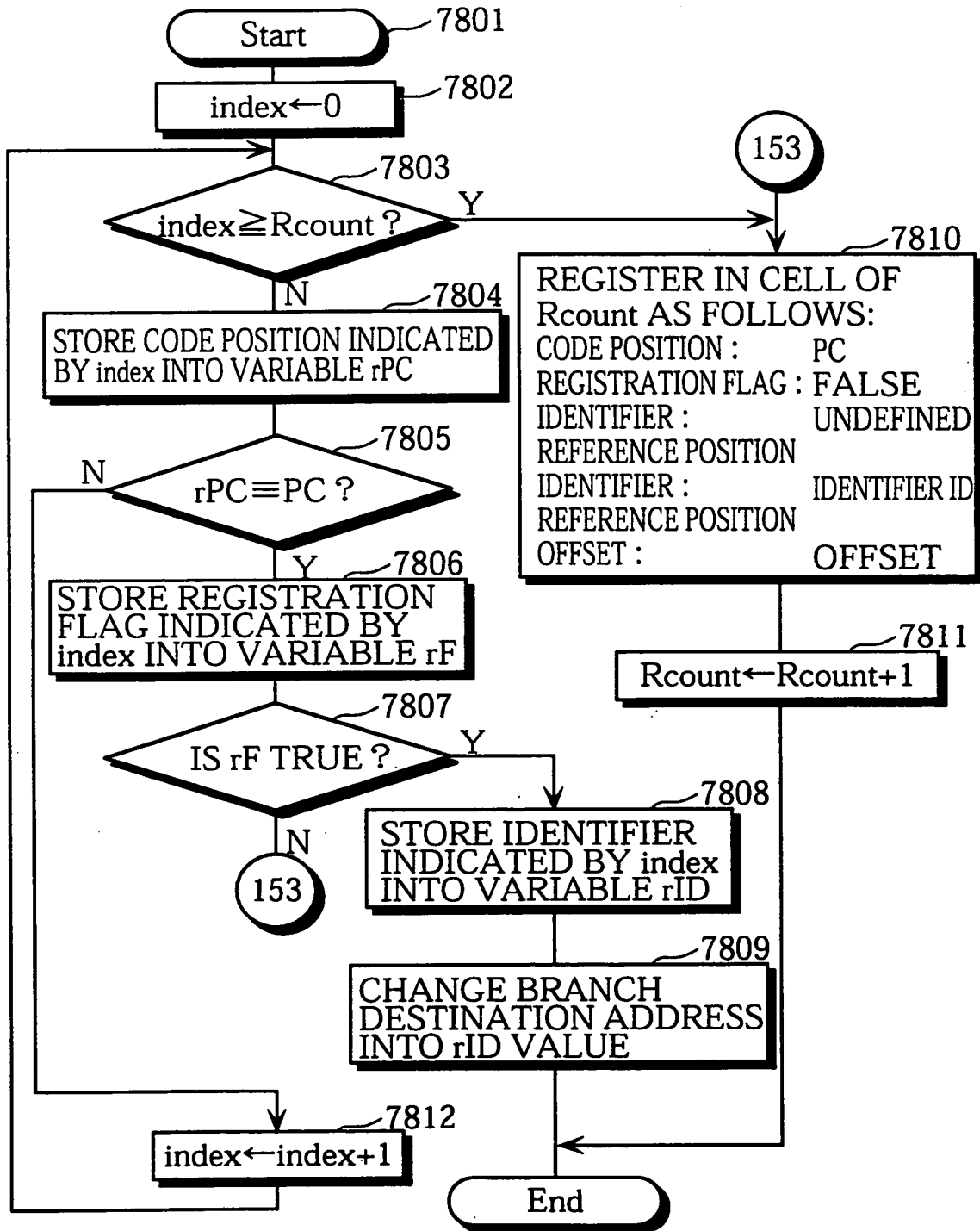


FIG. 83

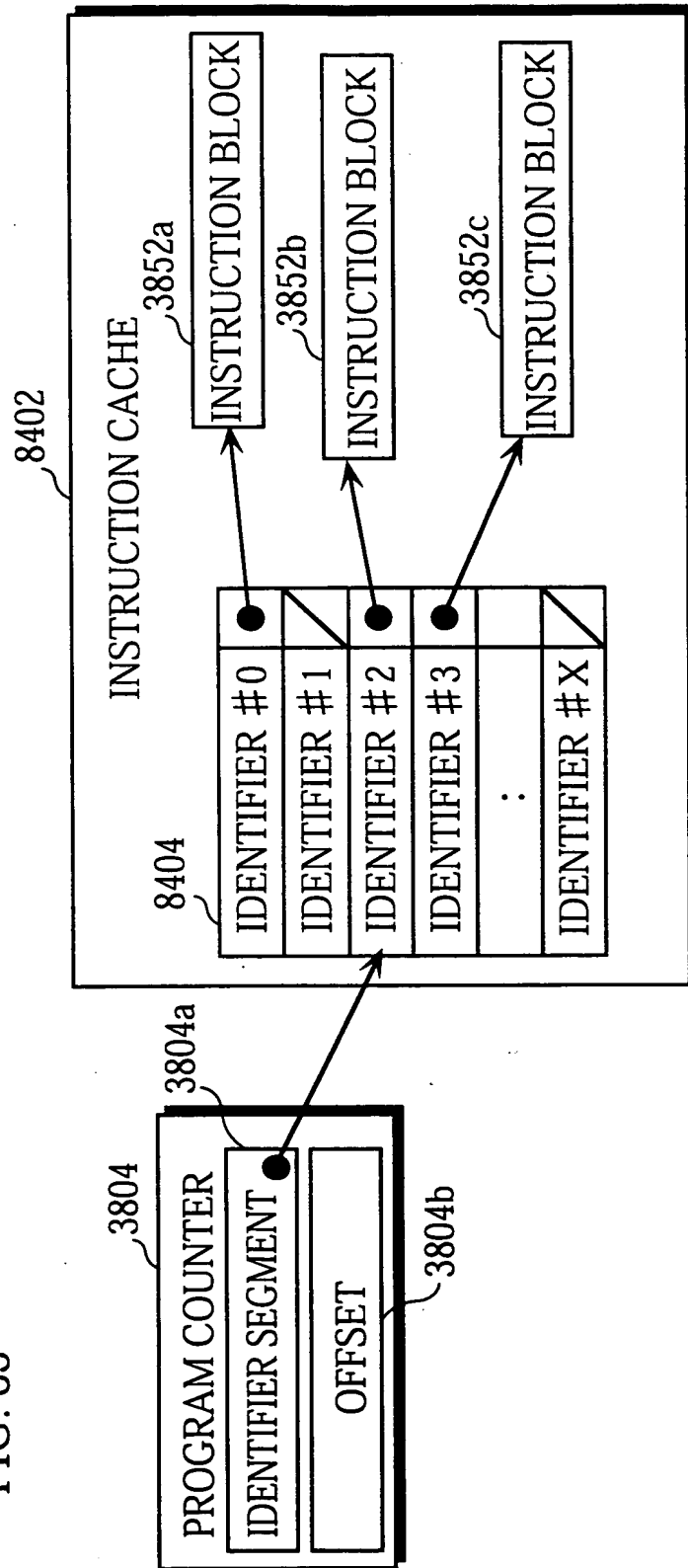


FIG. 84

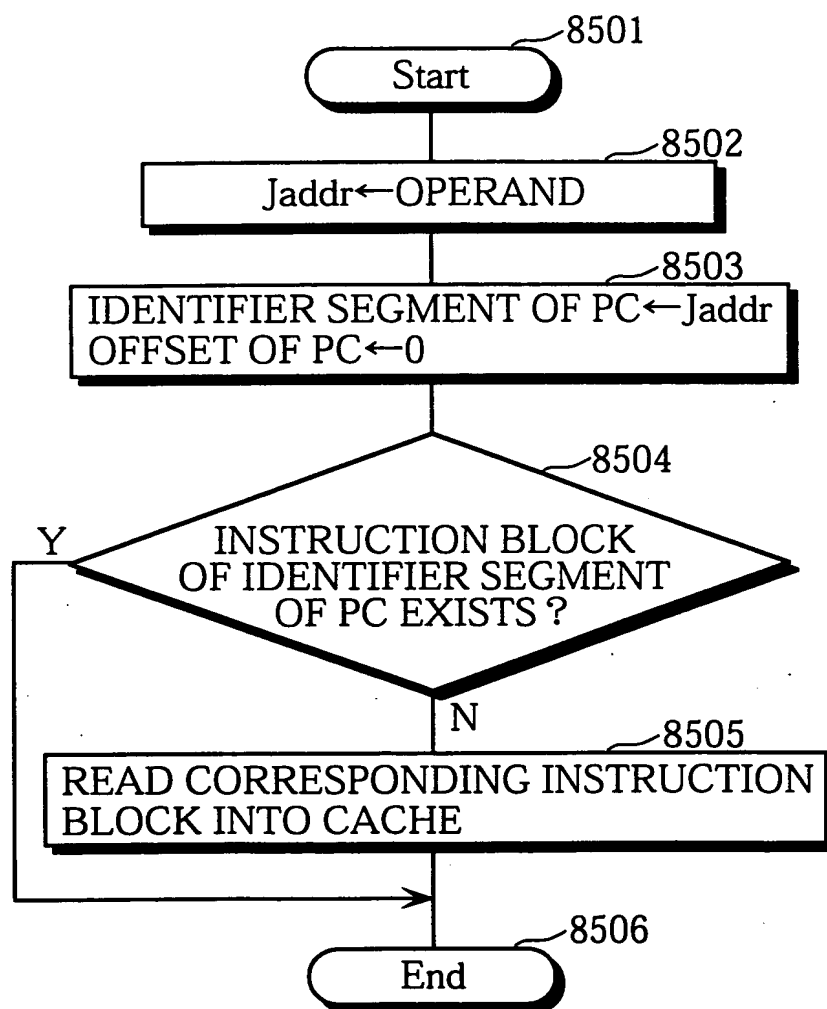


FIG. 85

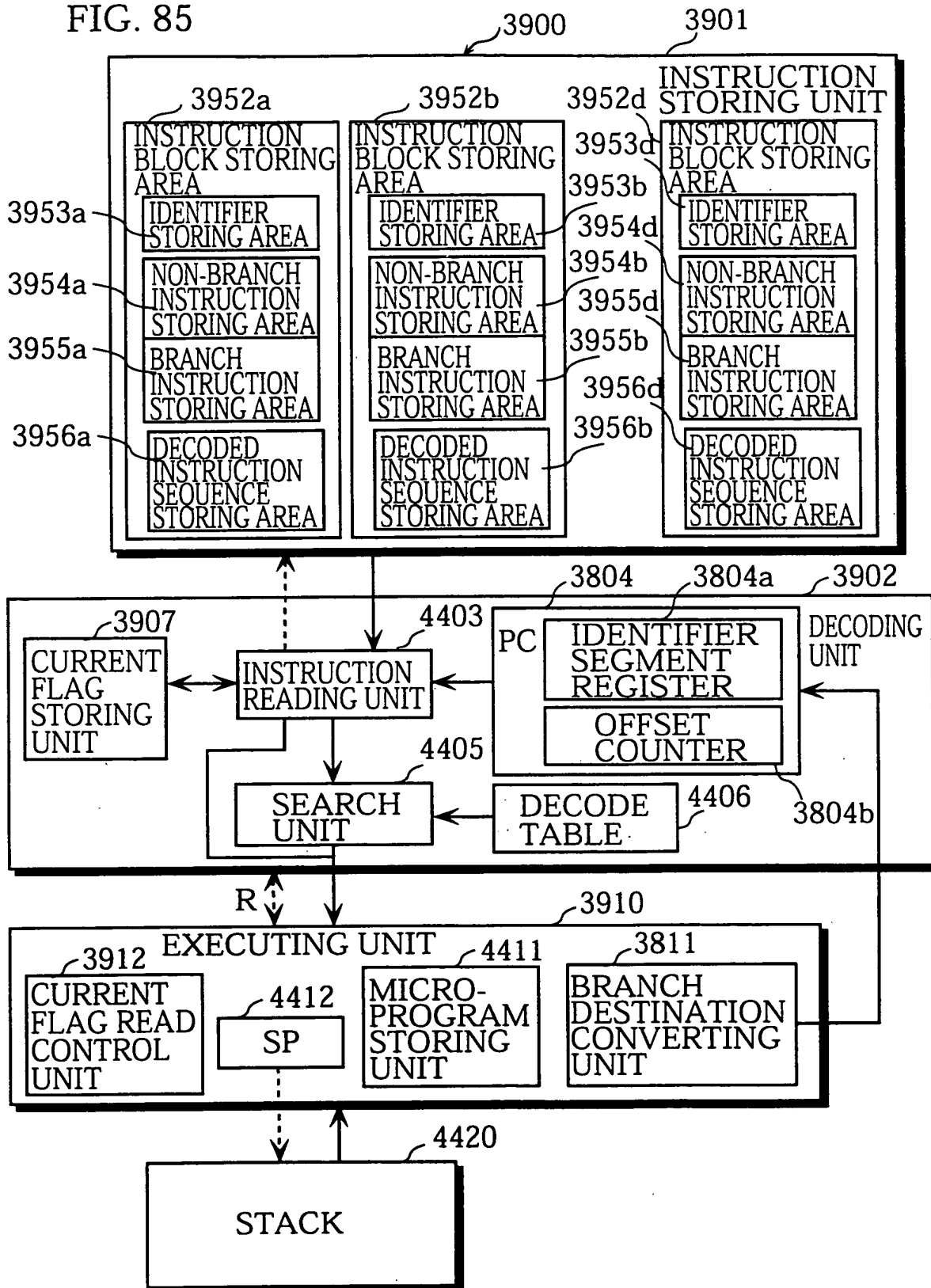


FIG. 86A

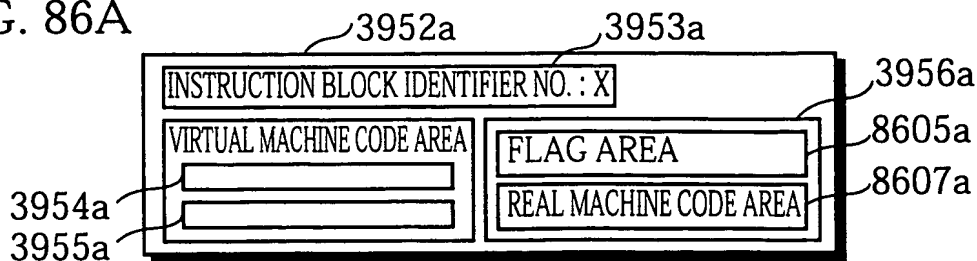


FIG. 86B

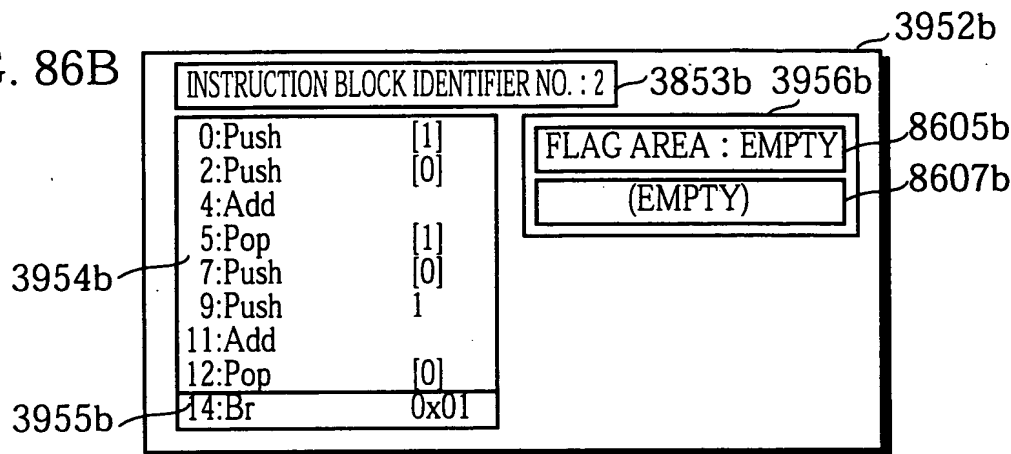


FIG. 86C

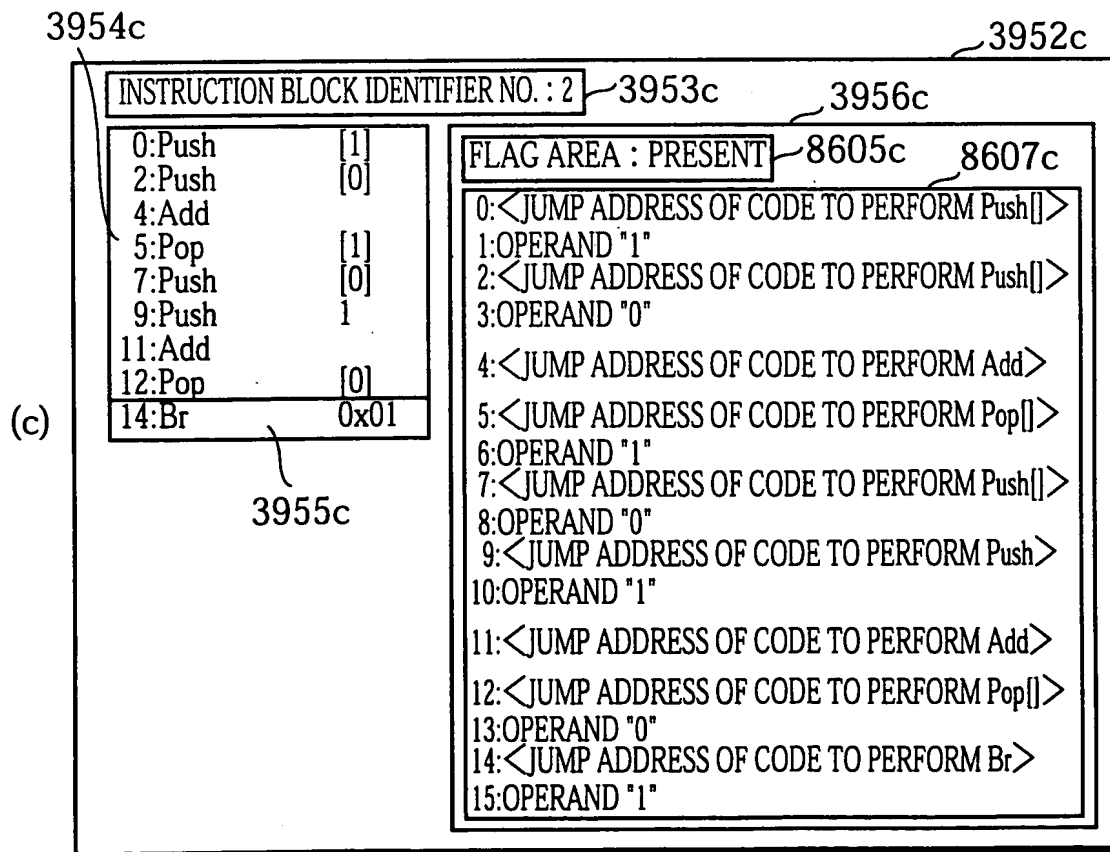


FIG. 87

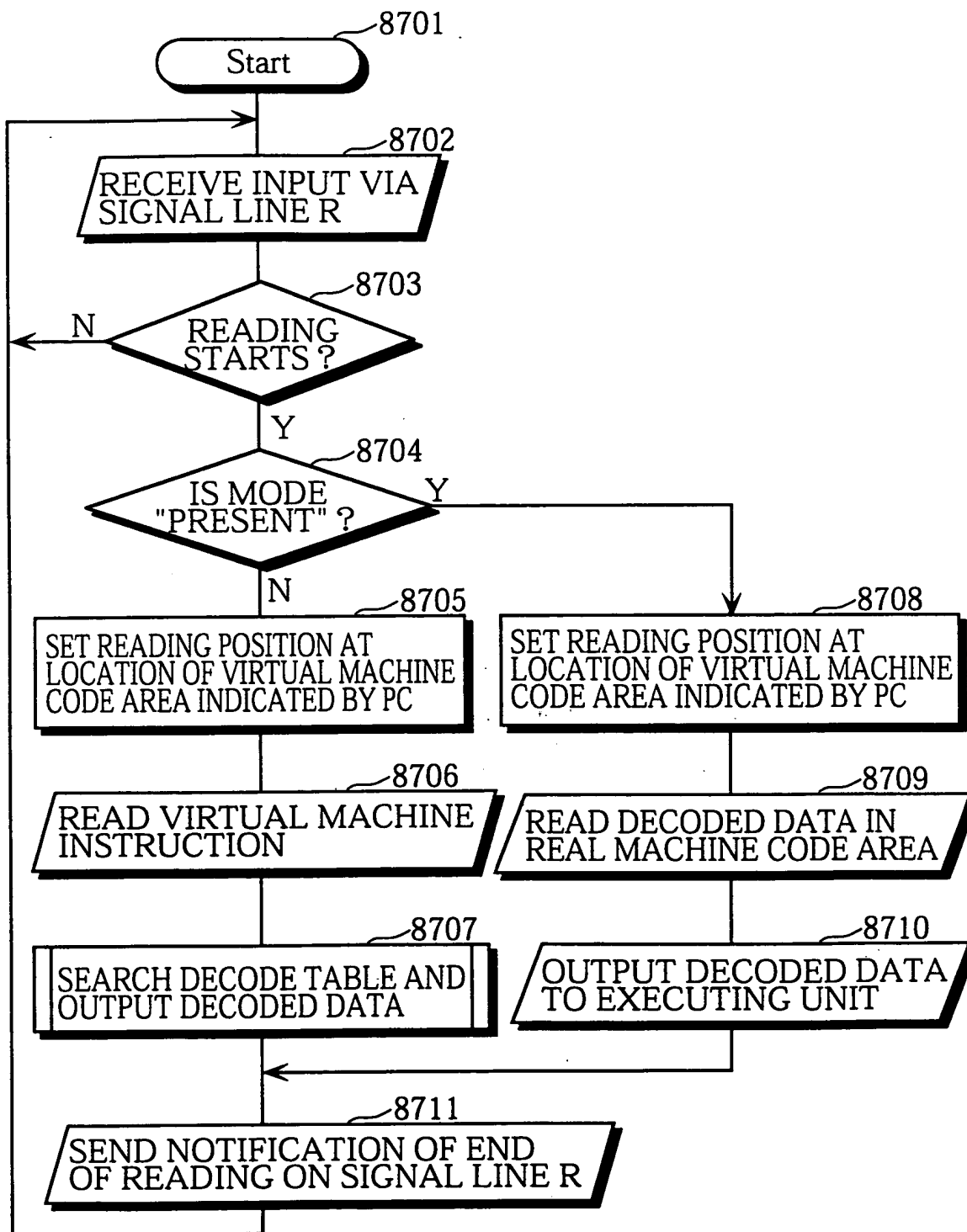




FIG. 88

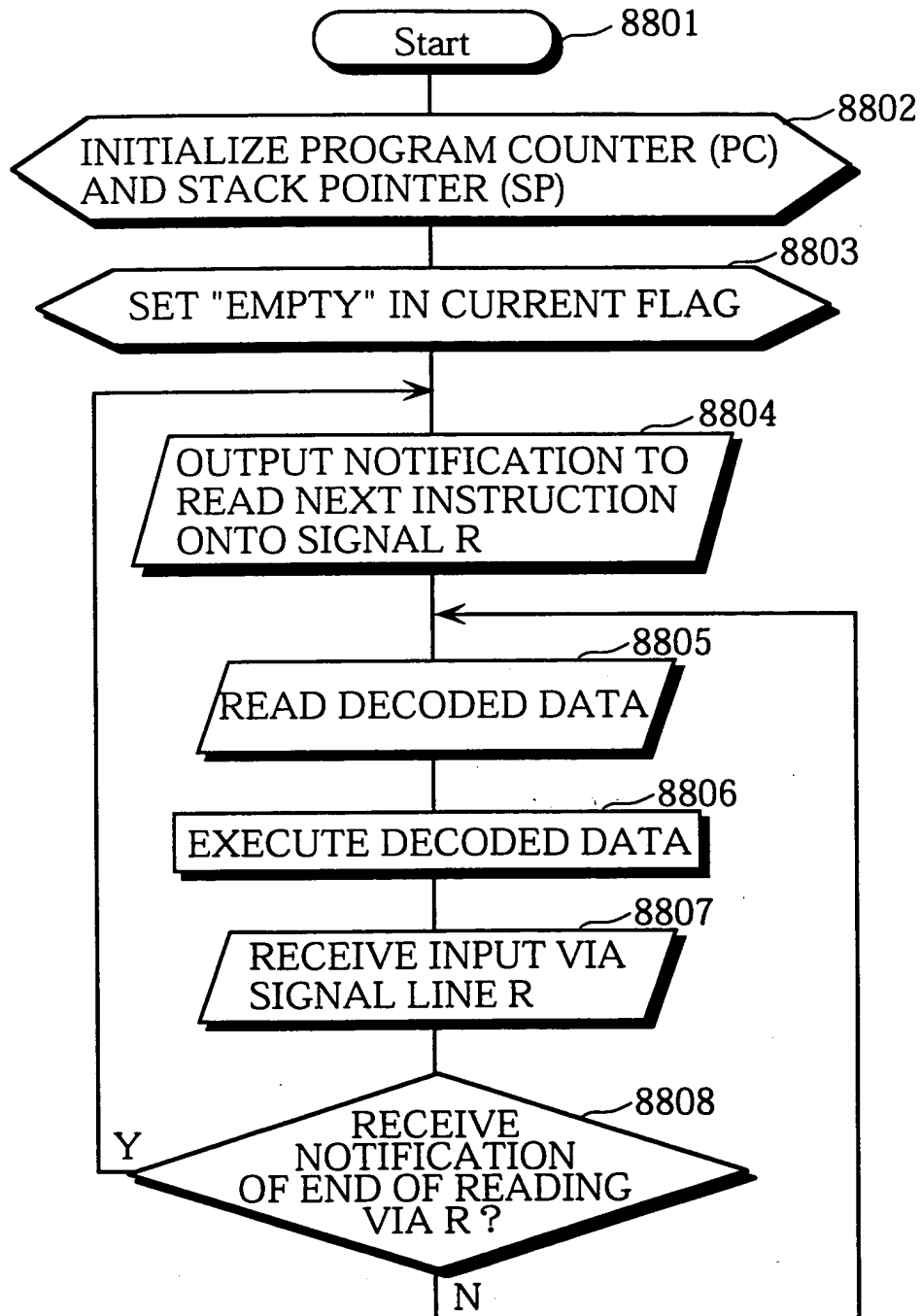




FIG. 89

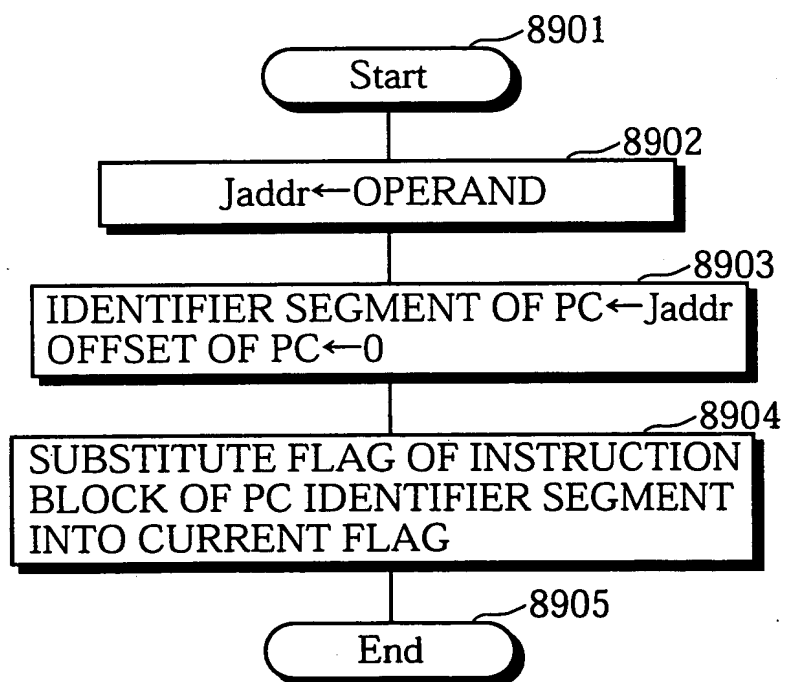


FIG. 90

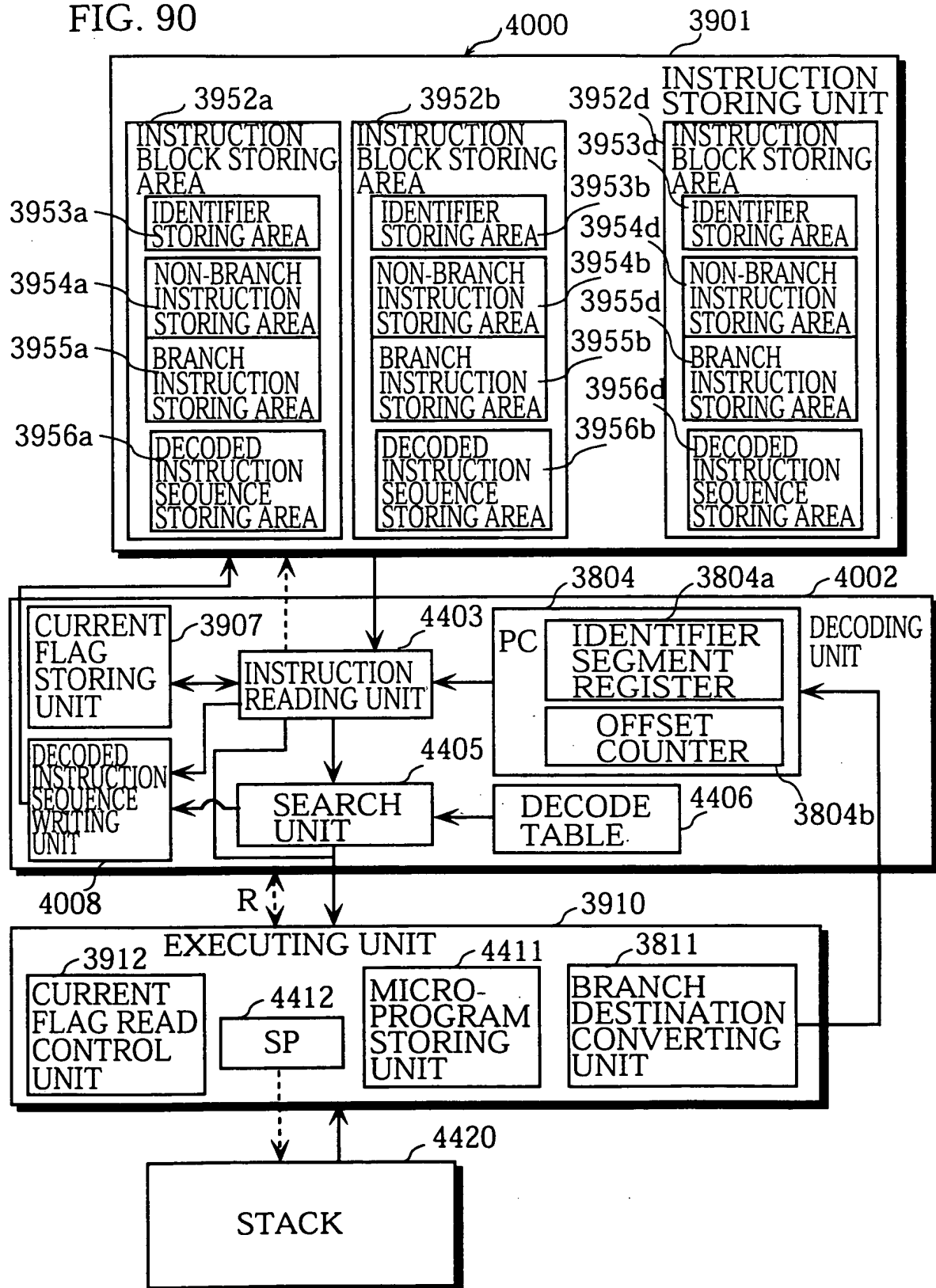


FIG. 91

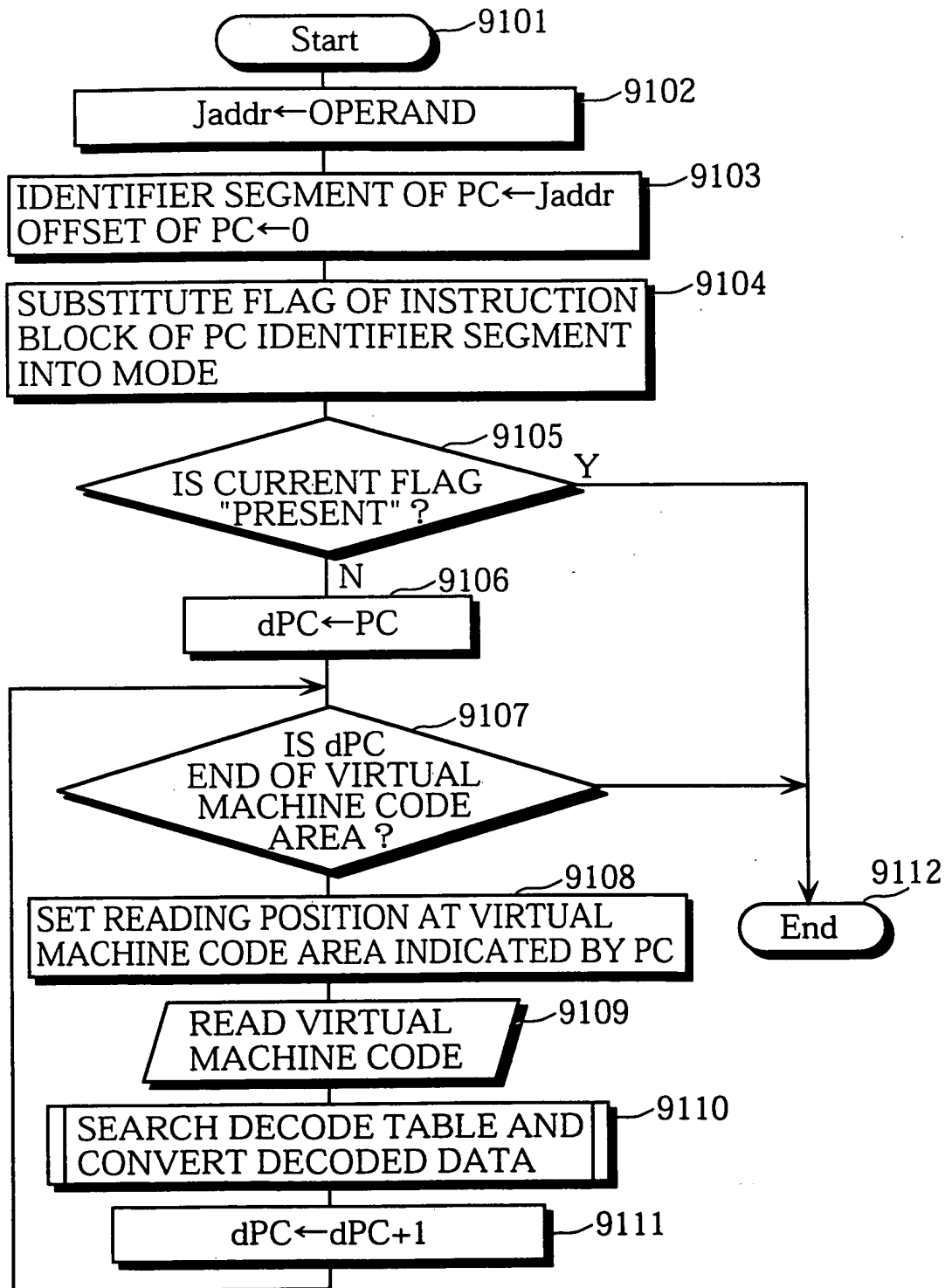


FIG. 92

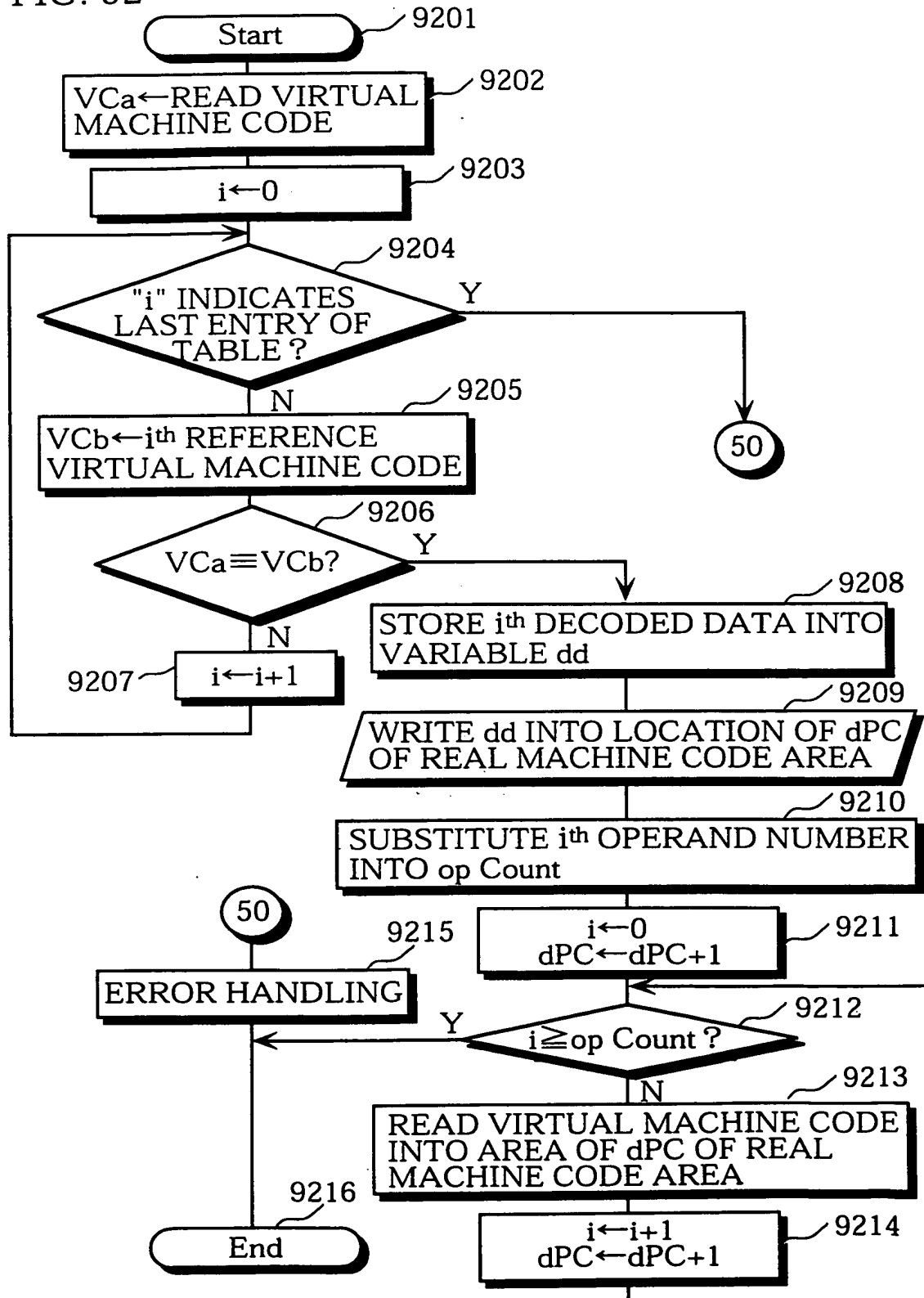




FIG. 93

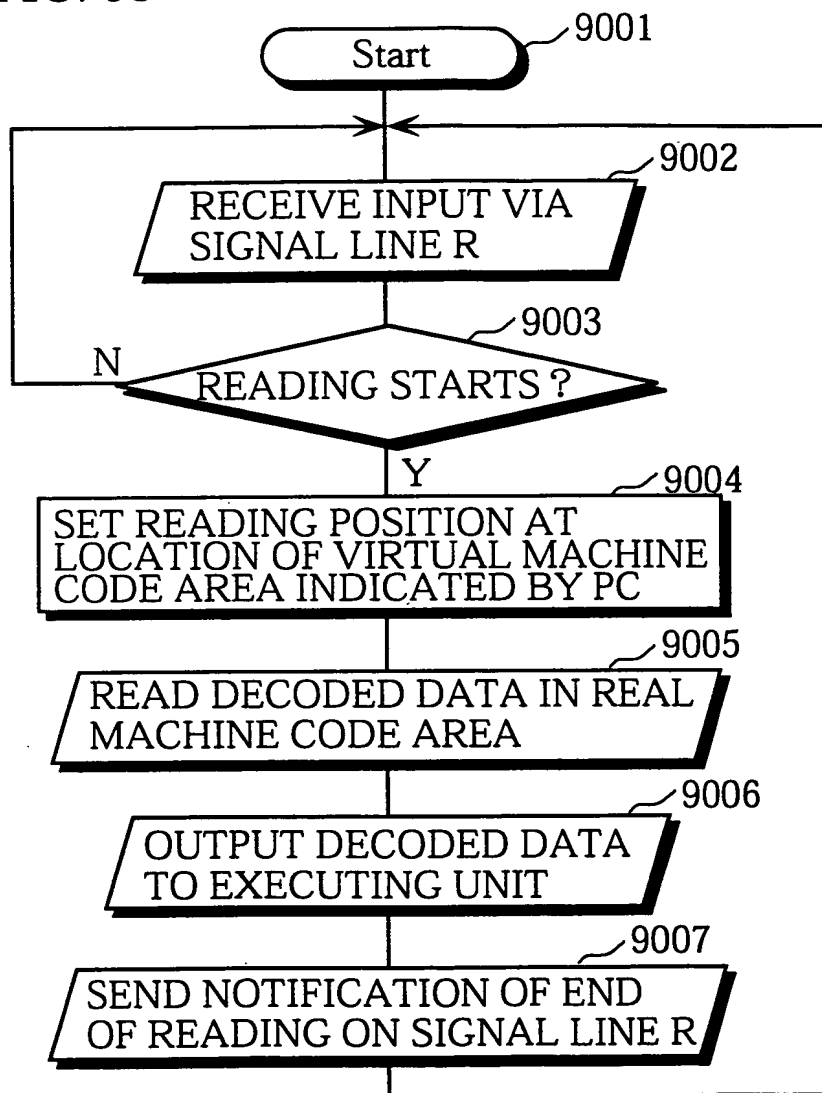


FIG. 94

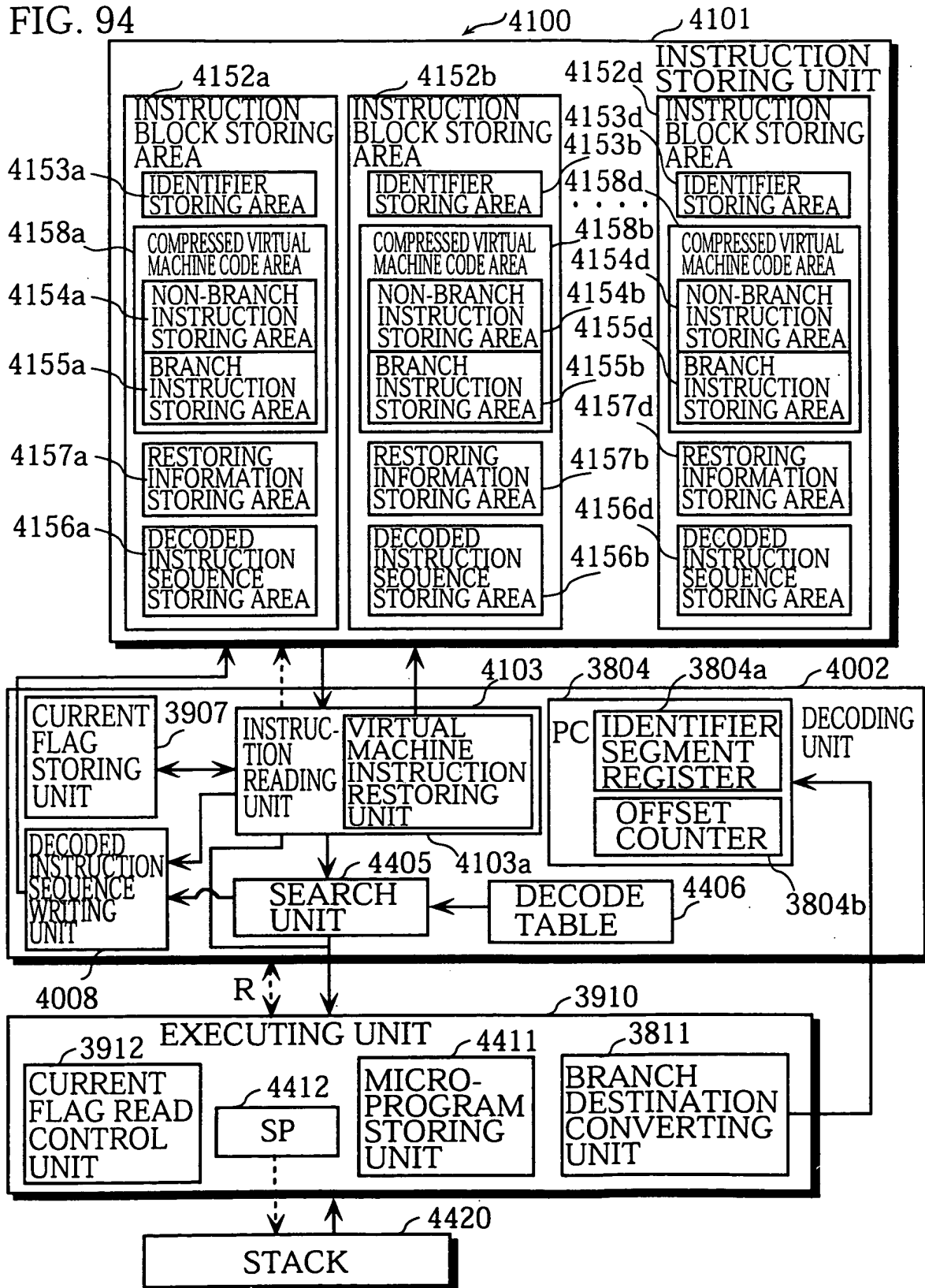


FIG. 95A

BIT SEQUENCE	VIRTUAL MACHINE INSTRUCTION
000	Push [0]
100	Push 0
101	Pop [0]
110	Pop [1]
111	Br 0x01
0011	Stop
0100	Add
0110	Brz 0x03
0111	Br 0x02
00100	Push [1]
00101	Push 1
01010	Push 10
01011	Sub

FIG. 95B

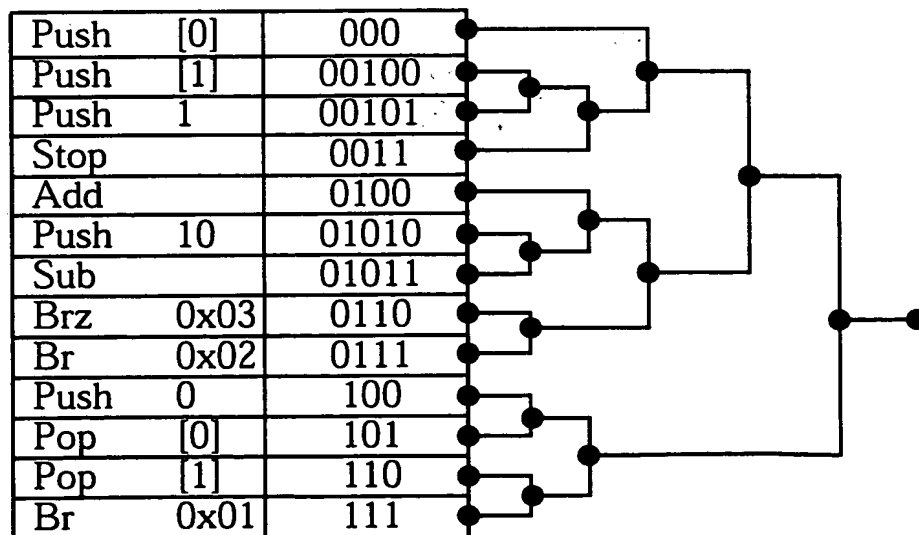


FIG. 96A

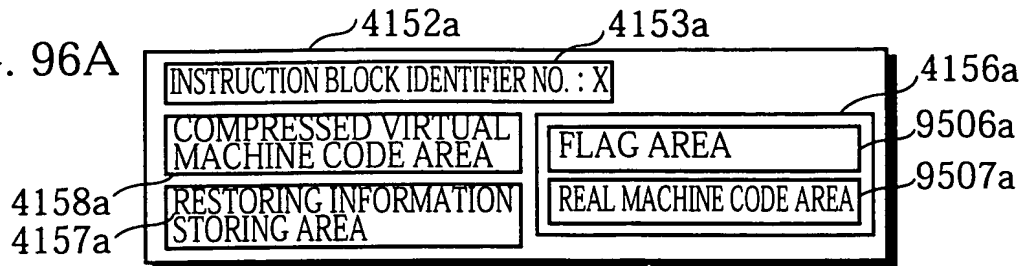


FIG. 96B

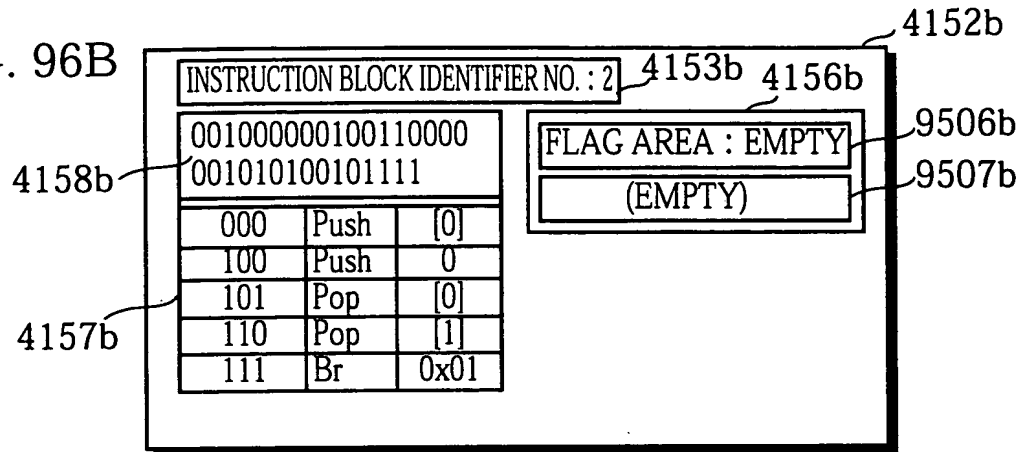


FIG. 96C

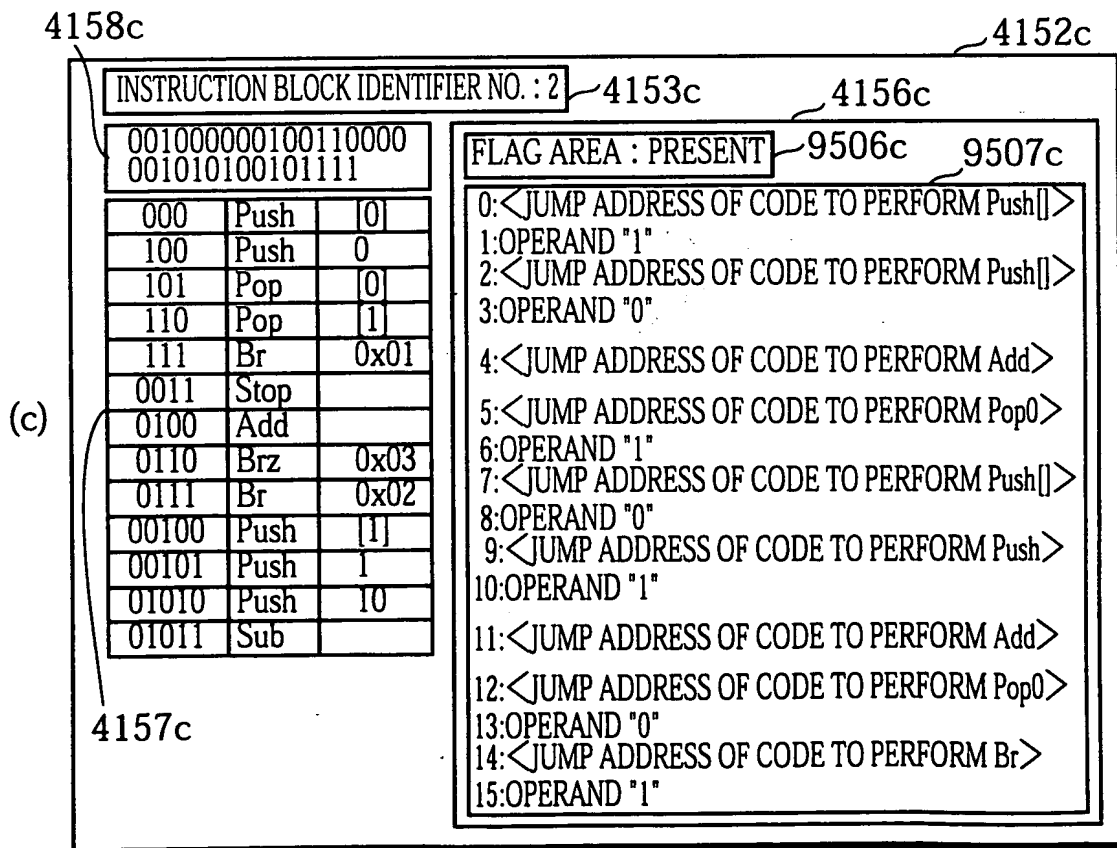


FIG. 97

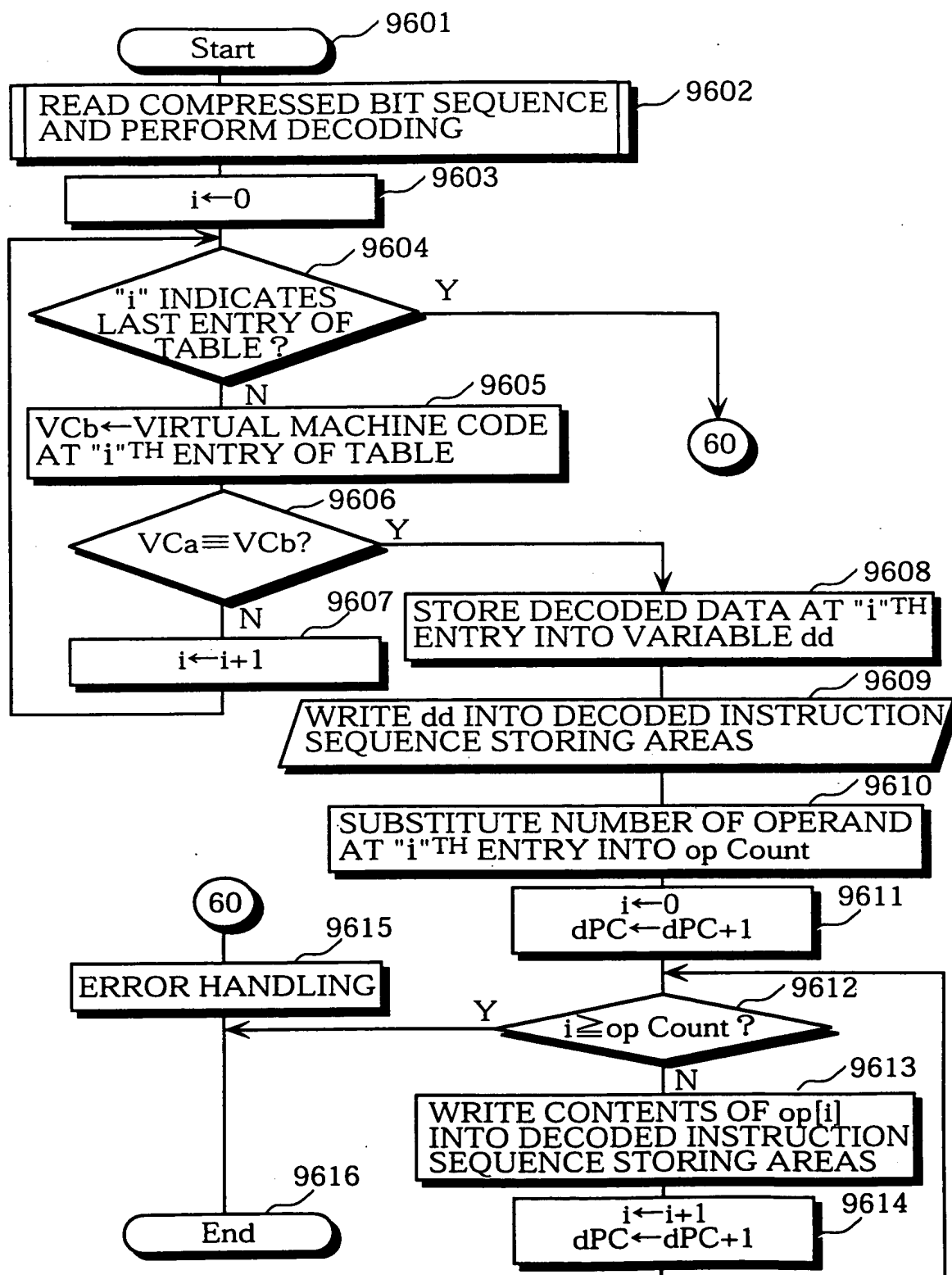


FIG. 98

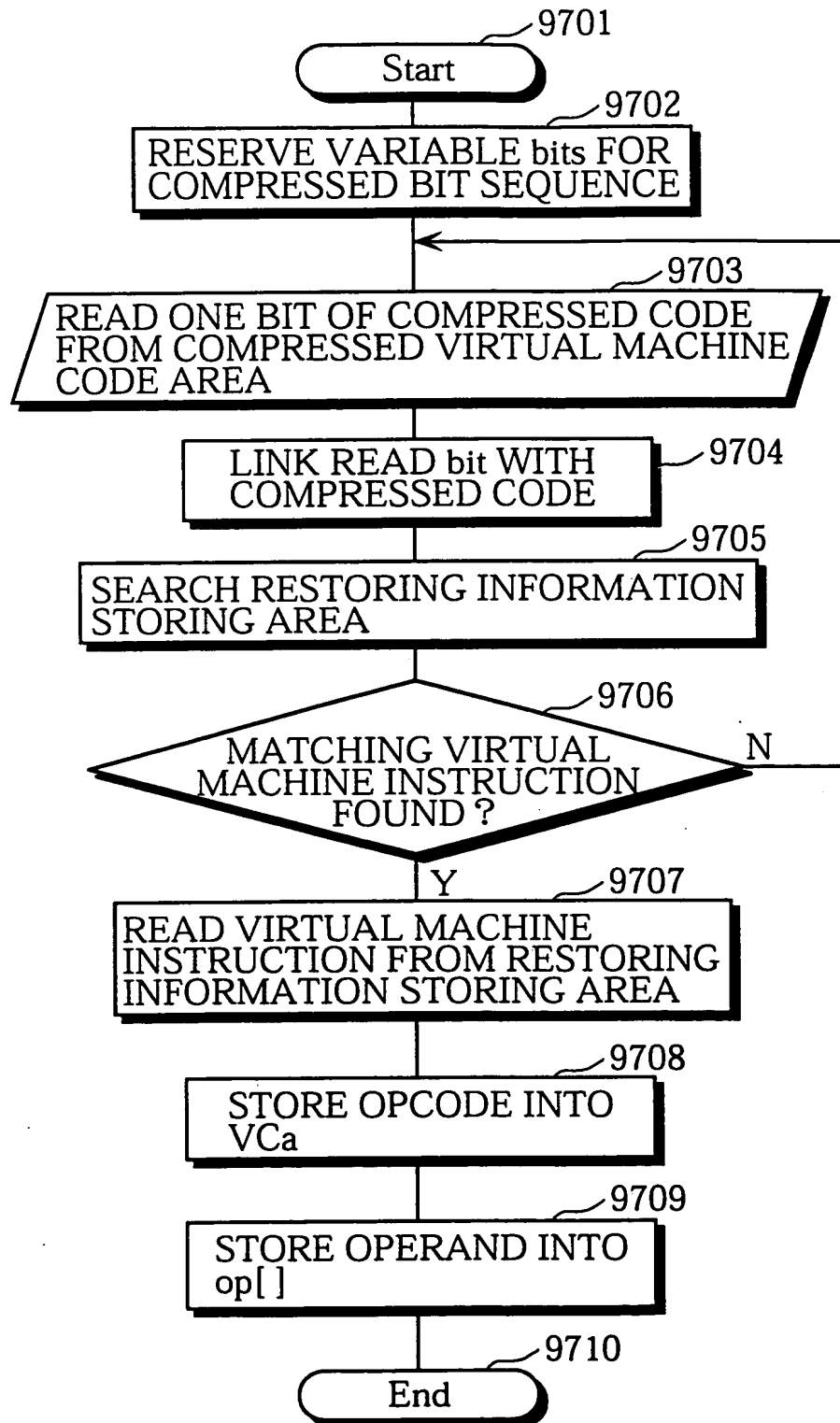


FIG. 99

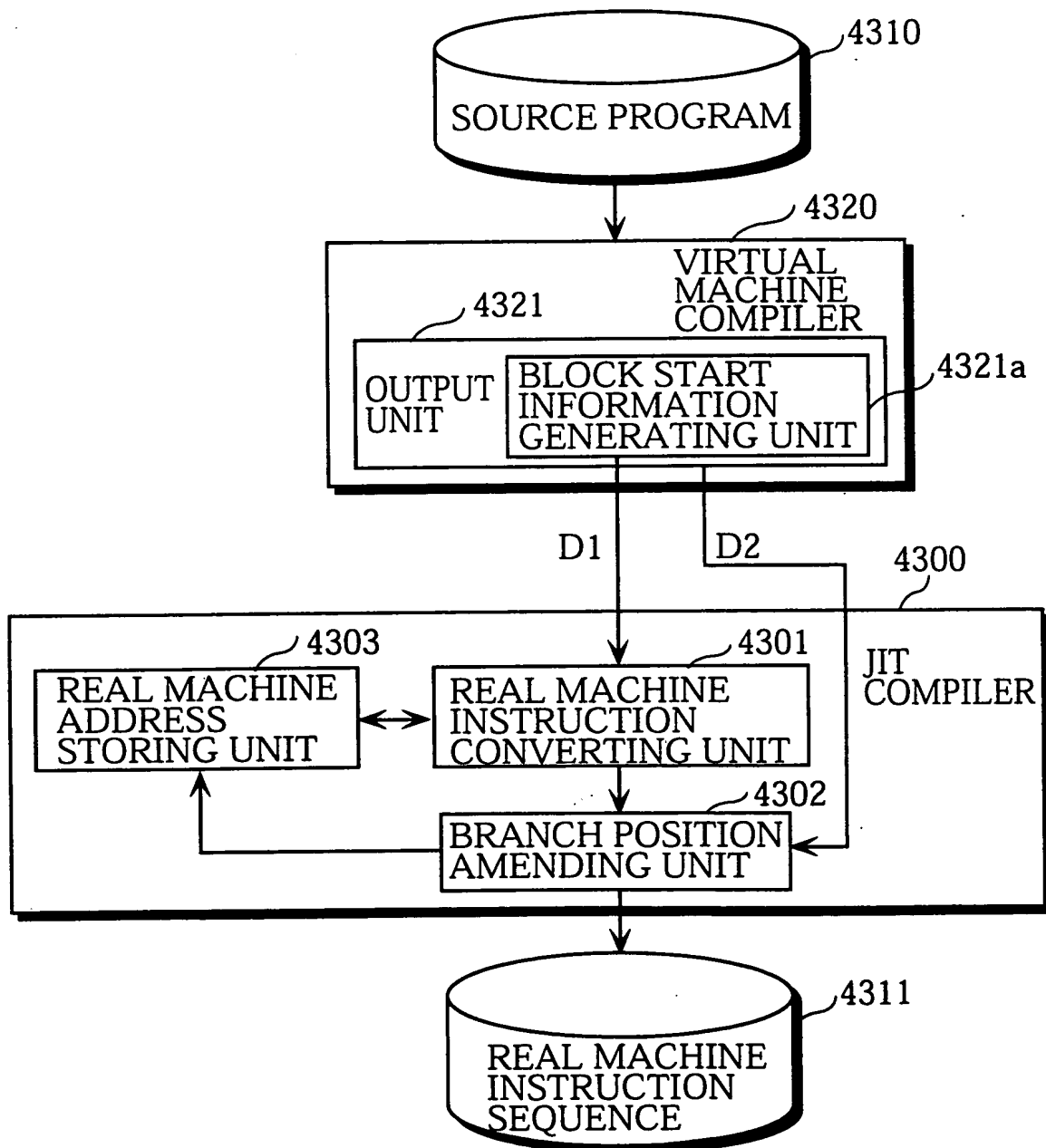


FIG. 100

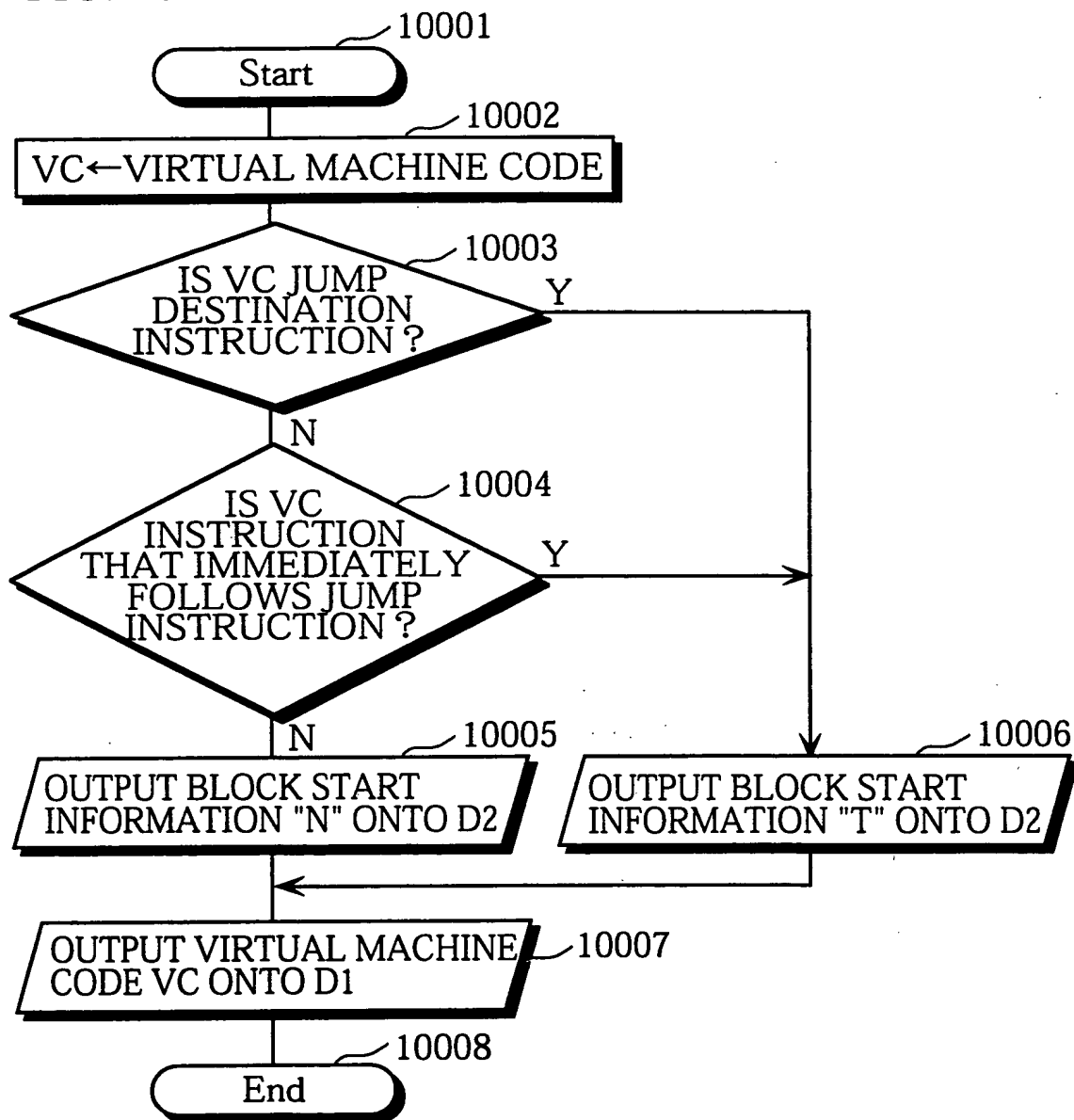


FIG. 101

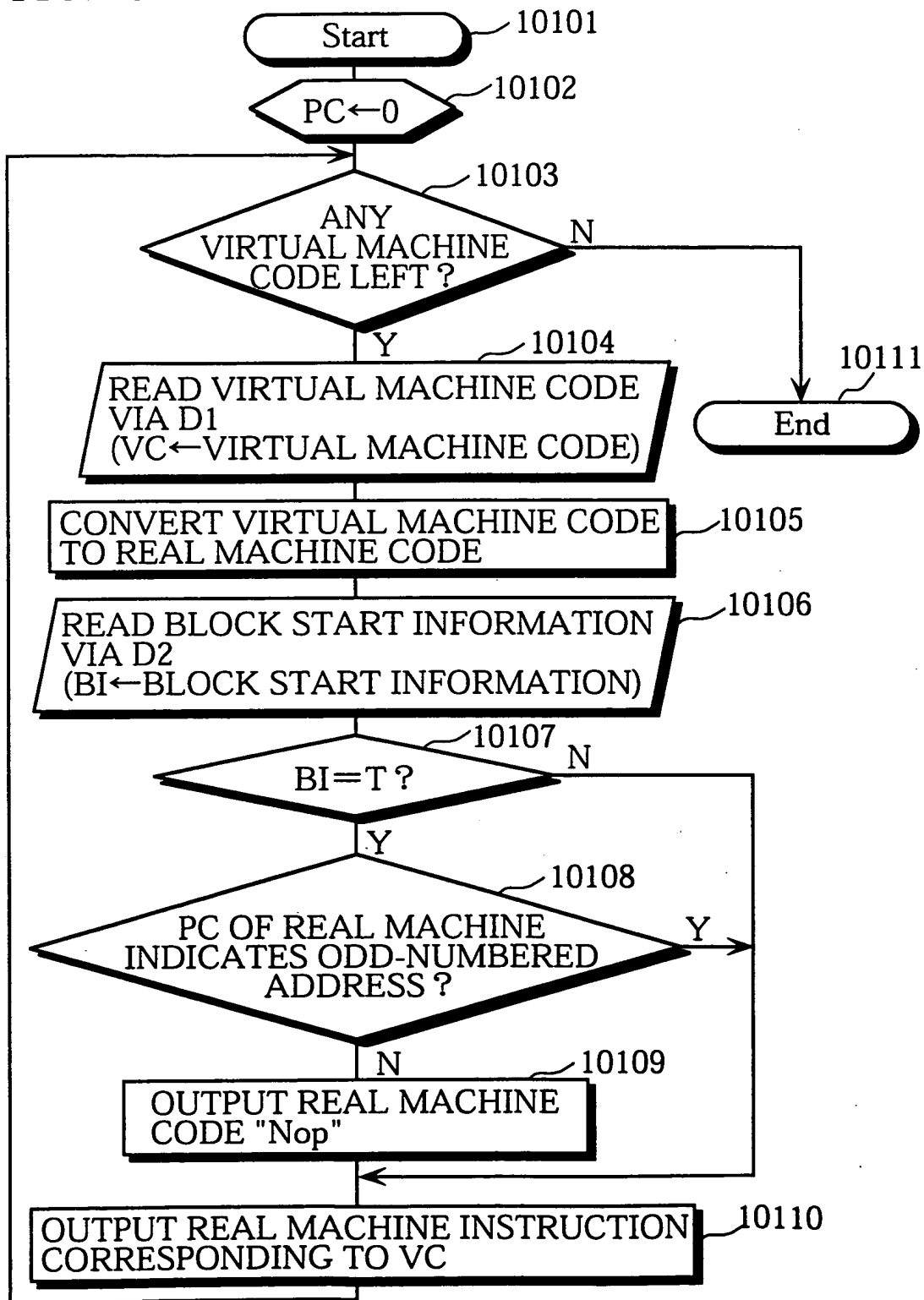


FIG. 102

ADDRESS	VIRTUL MACHINE CODE	REAL MACHINE CODE SIZE	CORRESPONDING REAL MACHINE CODE ADDRESS	BLOCK START INFORMATION	Nop OUTPUT
0	Push 0	4	0-3	T	-
2	Pop [0]	5	4-8	N	-
4	Push 0	4	9-12	N	-
6	Pop [1]	5	13-17	N	-
8	Push [0]	5	18-22	T	-
10	Push 10	4	23-26	N	-
12	Sub	3	27-29	N	-
13	Brz 31	5	30-34	N	Nop
15	Push [1]	5	36-40	T	-
17	Push [0]	5	41-45	N	-
19	Add	3	46-48	N	-
20	Pop [1]	5	49-53	N	-
22	Push [0]	5	54-58	N	-
24	Push 1	4	59-62	N	-
26	Add	3	63-65	N	-
27	Pop [0]	5	66-70	N	-
29	Br 8	3	71-73	N	-
31	Stop	2	74-75	T	-



FIG. 103

U/D	N/T	VIRTUAL MACHINE OPCODE	OPERAND(S)
-----	-----	---------------------------	------------